Shape-based Invariant Feature Extraction for Object Recognition

YANG Mingqiang[(1)], KPALMA Kidiyo[(2)], RONSIN Joseph[(2)]

[(1)]ISE, Shandong University, 250100, Jinan, China

[(2)]Université Européenne de Bretagne, France - INSA, IETR, UMR 6164, F-35708 RENNES

{yangmq@sdu.edu.cn, kidiyo.kpalma@insa-rennes.fr, joseph.ronsin@insa-rennes.fr}

Abstract:

The emergence of new technologies enables generating large quantity of digital information including images; this leads to an increasing number of generated digital images. Therefore it appears a necessity for automatic systems for image retrieval. These systems consist of techniques used for query specification and retrieval of images from an image collection. The most frequent and the most common means for image retrieval is the indexing using textual keywords. But for some special application domains and face to the huge quantity of images, keywords are no more sufficient or unpractical. Moreover, images are rich in content; so in order to overcome these mentioned difficulties, some approaches are proposed based on visual features derived directly from the content of the image: these are the content-based image retrieval (CBIR) approaches. They allow users to search the desired image by specifying image queries: a query can be an example, a sketch or visual features (e.g., colour, texture and shape). Once the features have been defined and extracted, the retrieval becomes a task of measuring similarity between image features. An important property of these features is to be invariant under various deformations that the observed image could undergo.

In this chapter, we will present a number of existing methods for CBIR applications. We will also describe some measures that are usually used for similarity measurement. At the end, and as an application example, we present a specific approach, that we are developing, to illustrate the topic by providing experimental results.

1 Introduction

Pattern recognition is the ultimate goal of most computer vision research. Shape feature extraction and representation are the bases of object recognition. It is also a research domain which plays an important role in many applications ranging from image analysis and pattern recognition, to computer graphics and computer anima-

tion. The feature extraction stage produces a representation of the content that is useful for shape matching. Usually the shape representation is kept as compact as possible for the purposes of efficient storage and retrieval and it integrates perceptual features that allow the human brain to discriminate between shapes. Efficient shape features must present some essential properties such as:

- **identifiability**: shapes which are found perceptually similar by human have the same features but different from the others,
- **translation, rotation and scale invariance**: the location, rotation and scaling changing of the shape must not affect the extracted features,
- **affine invariance**: the affine transform performs a linear mapping from 2D coordinates to other 2D coordinates that preserves the "straightness" and "parallelism" of lines. Affine transform can be constructed using sequences of translations, scales, flips, rotations and shears. The extracted features must be as invariant as possible with affine transforms.
- **noise resistance**: features must be as robust as possible against noise, i.e. they must be the same, in a given range, whichever be the strength of the noise that affects the pattern,
- **occultation resistance**: when some parts of a shape are occulted by other objects, the feature of the remaining part must not change, in a given range, compared to the original shape,
- **statistical independence**: two features must be statistically independent. This represents compactness of the representation,
- **reliability**: as long as one deals with the same pattern, the extracted features must remain the same.

In general, shape descriptor is some set of numbers that are produced to describe a given shape feature. A descriptor attempts to quantify shape in ways that agree with human intuition (or task-specific requirements). Good retrieval accuracy requires a shape descriptor to be able to effectively find perceptually similar shapes from a database. Usually, the descriptors are gathered under the form of a vector. Shape descriptors should meet the following requirements:

- **completeness**: the descriptors should be as complete as possible to represent the content of the information items,
- **compactness**: the descriptors should be represented and stored compactly. The size of descriptor vector must not be too large,
- **simplicity**: the computation of distance between descriptors should be simple; otherwise the execution time would be too long,
- **accessibility:** it describes how easy (or difficult) it is to compute a shape descriptor in terms of memory requirements and computation time,
- **large scope:** it indicates the extent of the class of shapes that can be described by the method,
- **uniqueness:** it indicates whether a one-to-one mapping exists between shapes and shape descriptors,
- **stability:** this describes how stable a shape descriptor is to “small” changes in shape.

Shape feature extraction and representation plays an important role in the following categories of applications:

- **shape retrieval**: searching for all shapes in a typically large database of shapes that are similar to a query shape. Usually all shapes within a given distance from the query are determined or at least the first few shapes that have the smallest distance.
- **shape recognition and classification**: determining whether a given shape matches a model sufficiently, or which one of representative class is the most similar,
- **shape alignment and registration**: transforming or translating one shape so that it best matches another shape, in whole or in part,
- **shape approximation and simplification**: constructing a shape from fewer elements (points, segments, triangles, etc.), that is still similar to the original.

To this end, many shape description and similarity measurement techniques have been developed in the past. A number of new techniques have been proposed in recent years, leading to three main classification methods c:

- **contour-based methods and region-based methods**: this is the most common and general classification and it is proposed by MPEG-7 which is a multimedia content description standard. It is based on the use of shape boundary points as opposed to shape interior points. Under each class, different methods are further divided into structural approaches and global approaches. This sub-class is based on whether the shape is represented as a whole or represented by segments/sections (primitives).
- **space domain and feature domain**: methods in space domain match shapes on point (or point feature) basis, while feature domain techniques match shapes on feature (vector) basis.
- **information preserving (IP) and non-information preserving (NIP)**: IP methods allow an accurate reconstruction of a shape from its descriptor, while NIP methods are only capable of partial ambiguous reconstruction. For object recognition purpose, IP is not a requirement.

Various algorithms and methods are documented in a vast literature. In this chapter, for sake of application conveniences, we reclassify them according to the **processing methods** i.e. the way the data of the shape are mathematically modelled and processed. The whole hierarchy of the classification is shown in figure 10.1.

Without being complete, we will describe and group a number of these methods together. So this chapter is organized as follows: section 2 presents 1D functions used in shape description. Section 3 presents some approaches for polygonal approximation of contours. Section 4 is dedicated to spatial interrelation features and section 5 presents shape moments. Sections 6 and 7 are, respectively, dedicated to scale space features and transform domain feature. Section 8 presents a summary table showing the properties of the methods. In order to illustrate this study, a practical example, based on a new shape descriptor, is presented in section 9.

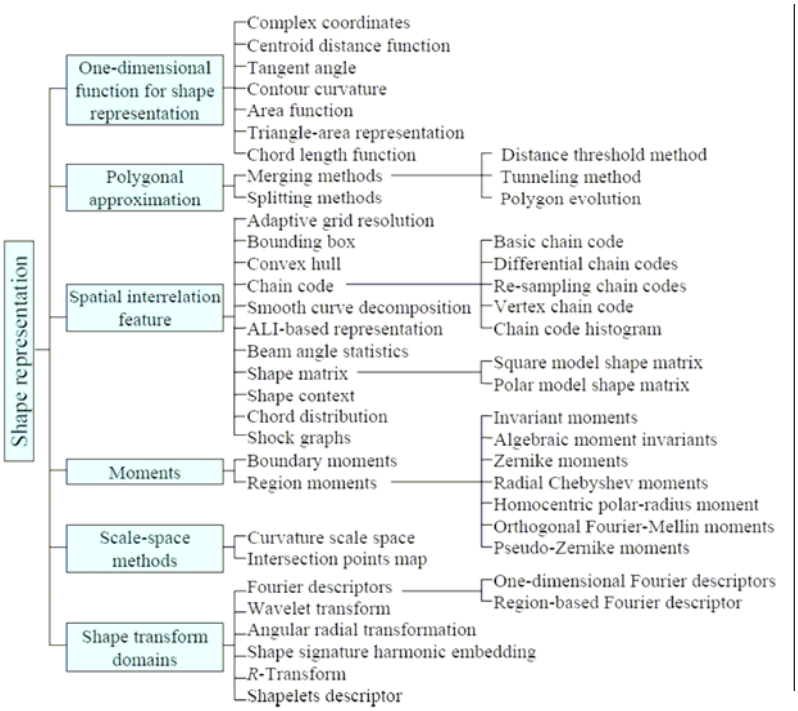


Figure 10.1: An overview of shape description techniques

2 One-dimensional function for shape representation

The one-dimensional function which is derived from shape boundary coordinates is also often called shape signature [22, 53]. The shape signature usually captures the perceptual feature of the shape [48]. Complex coordinates, centroid distance function, tangent angle (turning angles), curvature function, area function, triangle-area representation and chord length function are the commonly used shape signatures.

Shape signature can describe a shape all alone; it is also often used as a pre-processing to other feature extraction algorithms, for example, Fourier descriptors, wavelet description. In this section, the shape signatures are introduced.

2.1 Complex coordinates

A complex coordinates function is simply the complex number generated from the coordinates of boundary points, $P_n(x(n),y(n))$, $n \in [1,N]$:

$$z(n) = [x(n) - g_x] + i[y(n) - g_y] \tag{10.1}$$

where (g_x, g_y) is the centroid of the shape.

2.2 Centroid distance function

The centroid distance function r(n) is expressed by the distance of the boundary points from the centroid (g_x, g_y) of a shape, so that

$$r(n) = \sqrt{(x(n) - g_x)^2 + (y(n) - g_y)^2} \tag{10.2}$$

Due to the subtraction of centroid, which represents the position of the shape, from boundary coordinates, both complex coordinates and centroid distance representation are invariant to translation.

2.3 Tangent angle

The tangent angle function at a point $P_n(x(n); y(n))$ is defined by a tangential direction of a contour [54]:

$$\theta(n) = \theta_n = \arctan \frac{y(n) - y(n - \omega)}{x(n) - x(n - \omega)} \tag{10.3}$$

where ω represents a small window to calculate $\theta(n)$ more accurately, since every contour is a digital curve .

Tangent angle function has two problems. One is noise sensitivity. To decrease the effect of noise, the contour is filtered by a low-pass filter with appropriate bandwidth before calculating the tangent angle function. The other is discontinuity, due to the fact that the tangent angle function assumes values in a range of length 2π, usually in the interval of $[-\pi,\pi]$ or $[0,2\pi]$. Therefore θ_n in general contains discontinuities of size 2π. To overcome the discontinuity problem, with an arbitrary starting point, the cumulative angular function φ_n is defined as the angle differences between the tangent at any point P_n along the curve and the tangent at the starting point P_0 [30, 50]:

$$\varphi(n)=[\theta(n)-\theta(0)] \tag{10.4}$$

In order to be in accordance with human intuition that a circle is “shapeless”, assume $t=2\pi n/N$, then $\varphi(n)=\varphi(tN/2\pi)$. A periodic function is termed as the cumulative angular deviant function $\psi(t)$ and is defined as

$$y(t) = j(\frac{N}{2\pi}t) - t, \;\; t \in [0,2\pi] \tag{10.5}$$

where N is the total number of contour points.

In [25], the authors proposed a method based on tangent angle. It is called tangent space representation. A digital curve C simplified by polygon evolution is represented in the tangent space by the graph of a step function, where the x-axis represents the arc length coordinates of points in C and the y-axis represents the direction of the line segments in the decomposition of C. For example, figure 10.2

shows a digital curve and its step function representation in the tangent space.

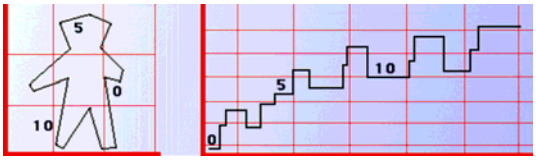


Figure 10.2: Digital curve and its step function representation in the tangent space

2.4 Contour curvature

Curvature is a very important boundary feature for human being to judge similarity between shapes. It also has salient perceptual characteristics and has proven to be very useful for shape recognition [47]. In order to use *K(n)* for shape representation, we quote the curvature function, *K(n)*, from [19, 32] as:

$$K(n) = \frac{\dot{x}(n)\ddot{y}(n) - \dot{y}(n)\ddot{x}(n)}{(\dot{x}(n)^2 - \dot{y}(n)^2)^{3/2}} \tag{10.6}$$

where $\dot{x}$ (or $\dot{y}$) and $\ddot{x}$ (or $\ddot{y}$) are, respectively, the first and second order derivatives of x (or y). Therefore, it is possible to compute the curvature of a planar curve from its parametric representation. If *n* is the normalized arc-length parameter *s*, then equation (10.6) can be written as:

$$K(s) = \dot{x}(s)\ddot{y}(s) - \dot{y}(s)\ddot{x}(s) \tag{10.7}$$

As given in equation (10.7), the curvature function is computed only from parametric derivatives, and, therefore, it is invariant under rotations and translations. However, the curvature measure is scale dependent, i.e., inversely proportional to the scale. A possible way to achieve scale independence is to normalize this measure by the mean absolute curvature, i.e.,

$$K'(s) = \frac{K(s)}{\frac{1}{N}\sum_{s=1}^{N}|K(s)|} \tag{10.8}$$

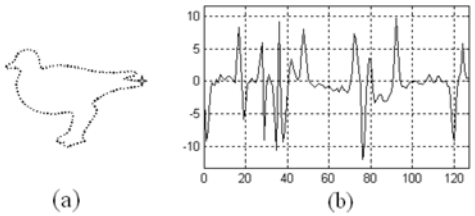
where *N* is the number of points on the normalized contour.

When the size of the curve is an important discriminative feature, the curvature should be used without the normalization; otherwise, for the purpose of scale-invariant shape analysis, the normalization should be performed by the following algorithm.

Let $P = \sum_{n=1}^{N} d_n$ be the perimeter of the curve and $L = \sum_{n=1}^{N} \sqrt{d_n}$, where d_n is the length of the chord between points p_n and p_{n+1}, *n=1, 2, ..., N-1*. An approximate arc-length parameterization based on the centripetal method is given by the following [19]:

$$s_k = s_{k-1} + \frac{P\sqrt{d_{k-1}}}{L}, k = 2,3,..., N \tag{10.9}$$

with s_1=0. Starting from an arbitrary point and following the contour clockwise, we compute the curvature at each interpolated point using equation (10.7). Figure 10.3 is an example of curvature function. Clearly, as a descriptor, the curvature function can distinguish different shapes.



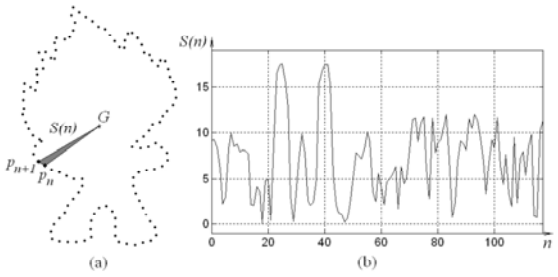
(a) Contours normalized to 128 points; the dots marked star are the starting points on the contours; (b) curvature functions; the curvature is computed clockwise.

Figure 10.3: Curvature function

Convex and concave vertices will imply negative and positive values, respectively (the opposite is verified for counter clockwise sense).

2.5 Area function

When the boundary points change along the shape boundary, the area of the triangle formed by two successive boundary points and the centre of gravity also changes. This forms an area function which can be exploited as shape representation. Figure 10.4 shows an example where $S(n)$ is the area between the successive boundary points P_n, P_{n+1} and centre of gravity G.



(a) Original contour; (b) the area function of (a).

Figure 10.4: Area function

The area function is linear under affine transform. However, this linearity only works for shape sampled at its same vertices.

2.6 Triangle-area representation

The triangle-area representation (TAR) signature is computed from the area of the triangles formed by the points on the shape boundary [2, 3]. The curvature of the contour point (x_n,y_n) is measured using the *TAR* function defined as follows:

For each three consecutive points $P_{n-t_s}(x_{n-t_s}, y_{n-t_s})$, $P_n(x_n, y_n)$, and $P_{n+t_s}(x_{n+t_s}, y_{n+t_s})$, where $n \in [1, N]$ and $t_s \in [1, N/2-1]$, N is even the signed area of the triangle formed by these points is given by:

$$TAR(n,t_s) = \frac{1}{2}\begin{vmatrix} x_{n-t_s} & y_{n-t_s} & 1 \\ x_n & y_n & 1 \\ x_{n+t_s} & y_{n+t_s} & 1 \end{vmatrix} \tag{10.10}$$

when the contour is traversed in counter clockwise direction, positive, negative and zero values of TAR mean convex, concave and straight-line points, respectively. Figure 10.5 shows these three types of the triangle areas and the complete TAR signature for the hammer shape.

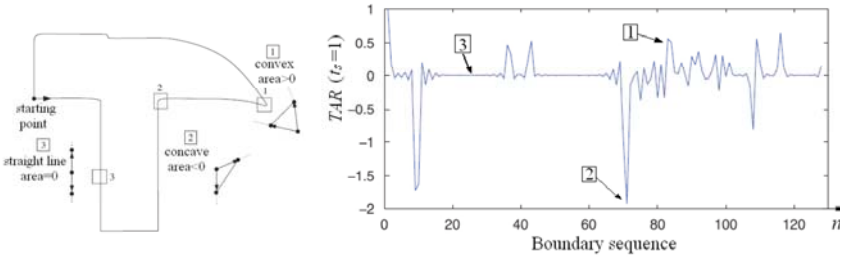


Figure 10.5: Three different types of the triangle-area values and the TAR signature for the hammer shape

By increasing the length of the triangle sides, i.e., considering farther points, the equation 10.10 will represent longer variations along the contour. The TARs with different triangle sides can be regarded as different scale space functions. The total TARs, $t_s \in [1, N/2-1]$, compose a multi-scale space TAR. In [3], authors show that the multi-scale space TAR is relatively invariant to the affine transform and robust to non-rigid transform.

2.7 Chord length function

The chord length function is derived from shape boundary without using any reference point. For each boundary point *p*, its chord length function is the shortest distance between *p* and another boundary point *p'* such that line *pp'* is perpendicular to the tangent vector at *p* [53].

The chord length function is invariant to translation and it overcomes the biased reference point (which means the centroid is often biased by boundary noise or de-

fections) problems. However, it is very sensitive to noise, there may be drastic burst in the signature of even smoothed shape boundary.

2.8 Discussions

A shape signature represents a shape by a 1-D function derived from shape contour. To obtain the translation invariant property, they are usually defined by relative values. To obtain the scale invariant property, normalization is necessary. In order to compensate for orientation changes, shift matching is needed to find the best matching between two shapes. Having regard to occultation, Tangent angle, Contour curvature and Triangle-area representation have invariance property. In addition, shape signatures are computationally simple.

Shape signatures are sensitive to noise, and slight changes in the boundary can cause large errors in matching. Therefore, it is undesirable to directly describe shape using a shape signature. Further processing is necessary to increase its robustness and reduce the matching load. For example, a shape signature can be simplified by quantizing the signature into a signature histogram, which is rotationally invariant.

3 Polygonal approximation

Polygonal approximation can be set to ignore the minor variations along the edge, and instead capture the overall shape. This is useful because it reduces the effects of discrete pixelization of the contour. In general, there are two methods to realize it. One is merging, the other is splitting [18].

3.1 Merging methods

Merging methods add successive pixels to a line segment if each new pixel that is added does not cause the segment to deviate too much from a straight line.

3.1.1 Distance threshold method

Choose one point as a starting point, on the contour. For each new point that we add, let a line go from the starting point to this new point. Then, we compute the squared error for every point along the segment/line. If the error exceeds some threshold, we keep the line from the start point to the previous point and start a new line. In practice, the most of practical error measures in use are based on distance between vertices of the input curve and the approximated linear segment [62]. The distance $d_k(i,j)$ from curve vertex $P_k(x_k,y_k)$ to the corresponding approximated linear segment defined by $P_i(x_i,y_i)$ and $P_j(x_j,y_j)$ is as follows (and illustrated in figure 10.6):

$$d_k(i,j) = \frac{\left|(x_j - x_i)(y_i - y_k) - (x_i - x_k)(y_j - y_i)\right|}{\sqrt{(x_j - x_i)^2 + (y_j - y_i)^2}} \tag{10.11}$$

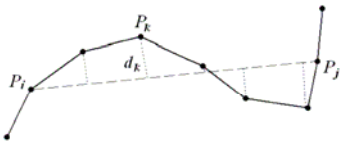


Figure 10.6: Illustration of the distance from a point on the boundary to a linear segment

3.1.2 Tunnelling method

If we have thick boundaries rather than single-pixel thick ones, we can still use a similar approach called tunnelling. Imagine that we are trying to lay straight rods along a curved tunnel, and that we want to use as few as possible. We can start at one point and lay a straight rod as long as possible. Eventually, the curvature of the "tunnel" won't let us go any further, so we lay one rod after another until we reach the end.

Both the distance threshold and tunnelling methods efficiently can do polygonal approximation. However, the great disadvantage is that the position of starting point will affect greatly the approximate polygon.

3.1.3 Polygon evolution

The basic idea of polygons evolution presented in [26] is very simple: in every evolution step, a pair of consecutive line segments (the line segment is the line between two consecutive vertices) s_1 and s_2 is substituted with a single line segment joining the endpoints of s_1 and s_2. The key property of this evolution is the order of the substitution. The substitution is done according to a relevance measure K given by

$$K(s_1, s_2) = \frac{\beta(s_1, s_2) l(s_1) l(s_2)}{l(s_1) + l(s_2)} \tag{10.12}$$

where $\beta(s_1, s_2)$ is the turn angle at the common vertex of segments s_1, s_2 and $l(\alpha)$ is the length of α, $\alpha = s_1$ or s_2, normalized with respect to the total length of a polygonal curve. The evolution algorithm assumes that vertices which are surrounded by segments with high values of $K(s_1, s_2)$ are more important than those with a low values (see figure 10.7 for illustration).

Figure 10.7: A few stages of polygon evolution according to a relevant measure

The curve evolution method achieves the task of shape simplification, i.e., the process of evolution compares the significance of vertices of the contour based on a relevance measure. Since any digital curve can be seen as a polygon without loss of information (with possibly a large number of vertices), it is sufficient to study evolutions of polygonal shapes for shape feature extraction.

3.2 Splitting methods

Splitting methods work by first drawing a line from one point on the boundary to another. Then, we compute the perpendicular distance from each point along the boundary segment to the line. If this exceeds some threshold, we break the line at the point of greatest distance. We then repeat the process recursively for each of the two new lines until we don't need to break any more. See figure 10.8 for an example.

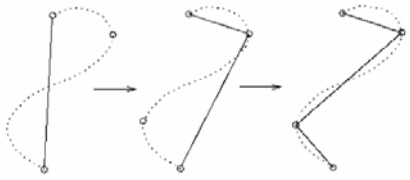


Figure 10.8: Splitting methods for polygonal approximation

This is sometimes known as the "fit and split" algorithm. For a closed contour, we can find the two points that lie farthest apart and fit two lines between them, one for one side and one for the other. Then, we can apply the recursive splitting procedure to each side.

3.3 Discussions

Polygonal approximation technique can be used as a simple method for contour representation and description. The polygon approximation has some interesting properties:

- it leads to simplification of shape complexity with no blurring effects,
- it leads to noise elimination,
- although irrelevant features vanish after polygonal approximation, there is no dislocation of relevant features,
- the remaining vertices on a contour do not change their positions after polygonal approximation.

Polygonal approximation technique can also be used as pre-processing method for further features extracting methods from a shape.

4 Spatial interrelation feature

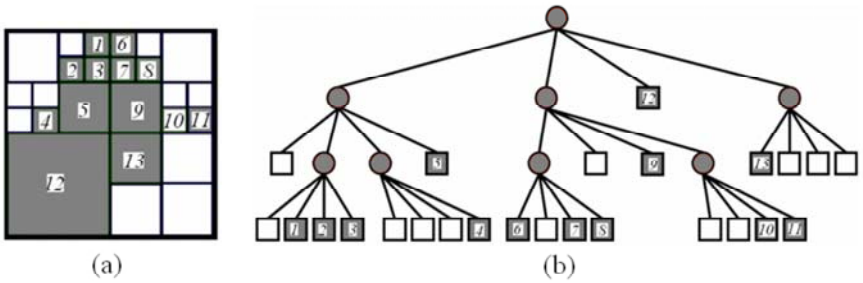
Spatial interrelation feature describes the region or the contour of shapes by observing and featuring the relations between their pixels or curves. In general, the representation is done by observing their geometric features: length, curvature, relative orientation and location, area, distance and so on.

4.1 Adaptive grid resolution

The adaptive grid resolution (AGR) scheme was proposed by [11]. In the AGR, a square grid that is just big enough to cover the entire shape is overlaid on it. A resolution of the grid cells varies from one portion to another according to the content of the portion of the shape. On the borders or the detail portion on the shape, the highest resolution, i.e. the smallest grid cells, are applied; on the other hand, in the coarse regions of the shape, lower resolution, i.e. the biggest grid cells, are applied.

To guarantee rotation invariance, it needs to reorient the shape into a unique common orientation. First, one has to find the major axis of the shape. The major axis defined as is the straight line segment joining the two points on the boundary farthest away from each other. Then rotate the shape so that its major axis is parallel to the *x*-axis.

One method to compute the AGR representation of a shape relies on a quad-tree decomposition on the bitmap representation of the shape [11]. The decomposition is based on successive subdivision of the bitmap into four equal-sized quadrants. If a bitmap-quadrant does not consist entirely of part of shape, it is recursively subdivided into smaller quadrants until we reach bitmap-quadrants, i.e., termination condition of the recursion is that the resolution reaches that one predefined: figure 10.9(a) shows an example of AGR.



(a) Adaptive Grid Resolution (AGR) image; (b) quad-tree decomposition of AGR.

Figure 10.9: Adaptive resolution representations

Each node in the quad-tree covers a square region of the bitmap. The level of the node in the quad-tree determines the size of the square. The internal nodes (shown by grey circles) represent “partially covered” regions; the leaf nodes shown by

white boxes represent regions with all 0*s* while the leaf nodes shown by black boxes represent regions with all 1*s*. The "all 1*s*" regions are used to represent the shape as shown on figure 10.9(b). Each rectangle can be described by 3 numbers: its centre coordinates $C = (C_x, C_y)$ and its size (i.e. side length) *S*. So each shape can be mapped to a point in 3*n*-dimensional space, where *n* is the number of the rectangles occupied by the shape region. Due to prior normalization, AGR representation is invariant under rotation, scaling and translation. It is also computationally simple.

4.2 Bounding box

Bounding box computes homeomorphisms between 2D lattices and its shapes. Unlike many other methods, this mapping is not restricted to simply connected shapes but applies to arbitrary topologies [7].

The minimum bounding rectangle or bounding box of *S* is denoted by *B(S)*; its width and height, are called *w* and *h*, respectively. An illustration of this procedure and its result is shown in figure 10.10.

(a) Compute the bounding box B(S) of a pixel set S; (b) subdivide S into n vertical slices; (c) compute the bounding box B(Sj) of each resulting pixel set S_j , where j=1, 2,…, n; (d) subdivide each B(Sj) into m horizontal slices; (e) compute the bounding box $B(S_{ij})$ of each resulting pixel set Sij , where i = 1, 2,…, m.

Figure 10.10: The five steps of bounding box splitting

Figure 10.11 shows the algorithm flowchart based on bounding box that divides a shape S into m (row)×n (column) parts. The output B is a set of bounding boxes.

If $v = (v_x, v_y)^T$ denotes the location of the bottom left corner of the initial bounding box of *S*, and $u_{ij} = (u_x^{ij}, u_y^{ij})^T$ denotes the centre of sample box $\mathbf{B}_{ij}$, then the coordinates

$$\begin{pmatrix} \mu_x^{ij} \\ \mu_y^{ij} \end{pmatrix} = \begin{pmatrix} (u_x^{ij} - v_x)/w \\ (u_y^{ij} - v_y)/h \end{pmatrix} \tag{10.13}$$

provide a scale invariant representation of *S*. Sampling *k* points of an *m*×*n* lattice therefore allows to represent *S* as a vector

$$r = [\mu_x^{i(1)j(1)}, \mu_y^{i(1)j(1)}, ..., \mu_x^{i(k)j(k)}, \mu_y^{i(k)j(k)}] \tag{10.14}$$

where $i(\alpha) < i(\beta)$ if $\alpha < \beta$ and likewise for the index *j*.

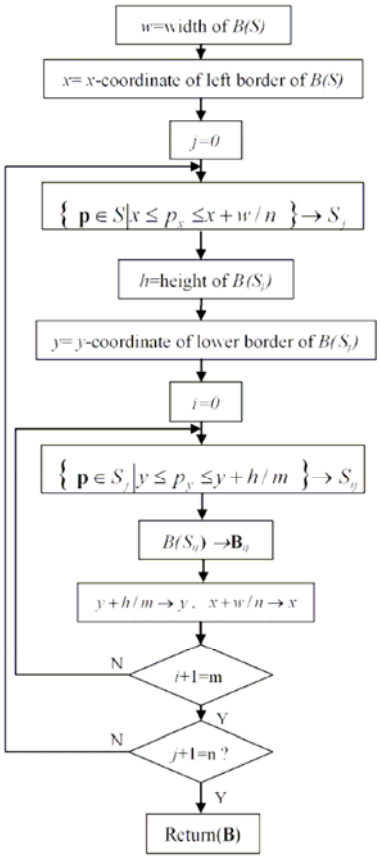


Figure 10.11: Flowchart of shape divided by bounding box

To represent each bounding box, one method consists of sampling partial points of the set of bounding boxes (see figure 10.12).

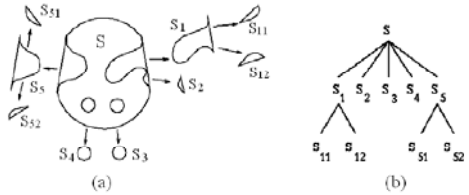
Figure 10.12: A sample points on lattice and examples of how it is mapped onto different shapes

Bounding box representation is a simple computational geometry approach to compute homeomorphisms between shapes and lattices. It is storage and time efficient. It is invariant to rotation, scaling and translation and also robust against noisy shape boundaries.

4.3 Convex hull

The approach is based on the fact that the shape is represented by a serie of convex hulls. The convex hull H of a region consists of its smallest convex region including it. In other words, for a region S, the convex hull $conv(S)$ is defined as the smallest convex set in R^2 containing S. In order to decrease the effect of noise, common practice is to first smooth a boundary prior to partitioning it.

The representation of the shape may then be obtained by a recursive process which results in a concavity tree (see figure 10.13). Each concavity can be described by its area, chord (the line connects the cut of the concavity) length, maximum curvature, distance from maximum curvature point to the chord. The matching between shapes becomes a string or a graph matching.



(a) Convex hull and its concavities; (b) concavity tree representation of convex hull.

Figure 10.13: Illustration of recursive process of convex hull

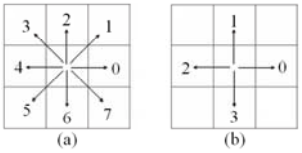
Convex hull representation has a high storage efficiency. It is invariant to rotation, scaling and translation and also robust against noisy shape boundaries (after filtering). However, extracting the robust convex hulls from the shape is where the shoe pinches. [14, 16] and [41] gave the boundary tracing method and morphological methods to achieve convex hulls respectively.

4.4 Chain code

Chain code is a common approach for representing different rasterized shapes as line-drawings, planar curves, or contours. Chain code describes an object by a sequence of unit-size line segments with a given orientation [51]. Chain code can be viewed as a connected sequence of straight-line segments with specified lengths and directions [28].

4.4.1 Basic chain code

Freeman [57] first introduced a chain code that describes the movement along a digital curve or a sequence of border pixels by using so-called 8-connectivity or 4-connectivity. The direction of each movement is encoded by the numbering scheme i=0, 1, …,7 or i=0, 1, 2, 3 denoting a counter-clockwise angle of $45^{o}\times i$ or $90^{o}\times i$ regarding the positive x-axis, as shown in figure 10.14.



(a) Chain code in eight directions (8-connectivity); (b) chain code in four directions (4-connectivity).

Figure 10.14: Basic chain code direction

By encoding relative, rather than absolute position of the contour, the basic chain code is translation invariant. We can match boundaries by comparing their chain codes, but with the two main problems: 1) it is very sensitive to noise; 2) it is not rotationally invariant. To solve these problems, differential chain codes (DCC) and resampling chain codes (RCC) were proposed.

DCC encodes differences in the successive directions. This can be computed by subtracting each element of the chain code from the previous one and taking the result modulo *n*, where *n* is the connectivity. This differencing process allows us to rotate the object in 90-degree increments and still compare the objects, but it doesn't get around the inherent sensitivity of chain codes to rotation on the discrete pixel grid.

RCC consists of re-sampling the boundary onto a coarser grid and then computing the chain codes of this coarser representation. This smoothes out small variations and noise but can help compensate for differences in chain-code length due to the pixel grid.

4.4.2 Vertex chain code (VCC)

To improve chain code efficiency, in [28] the authors proposed a chain code for shape representation according to VCC. An element of the VCC indicates the number of cell vertices, which are in touch with the bounding contour of the shape in that element's position. Only three elements "1", "2" and "3" can be used to represent the bounding contour of a shape composed of pixels in the rectangular grid. Figure 10.15 shows the elements of the VCC to represent a shape.

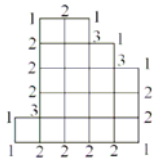


Figure 10.15: Vertex chain code

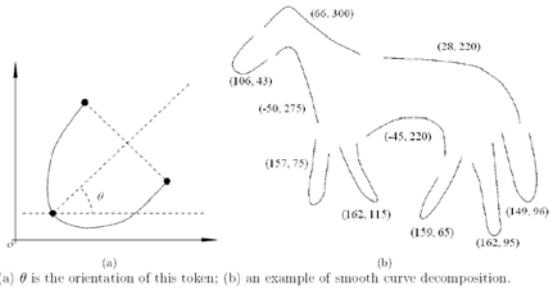
4.4.3 Chain code histogram (CCH)

Iivarinen and Visa have derived a CCH for object recognition [58]. The CCH is computed as $h_i=\#\{i \in M$, M is the range of chain code$\}$, $\#\{\alpha\}$ denotes getting the number of the value α. The CCH reflects the probabilities of different directions

present in a contour. If the chain code is used for matching it must be independent of the choice of the starting pixel in the sequence. The chain code usually has high dimensions and is sensitive to noise and any distortion. So, except for the CCH, the other chain code approaches are often used as contour representations, but not as contour attributes.

4.5 Smooth curve decomposition

In [9], the authors proposed smooth curve decomposition as shape descriptor. The segment between the curvature zero-crossing points from a Gaussian smoothed boundary are used to obtain primitives, called tokens. The feature for each token corresponds to its maximum curvature and its orientation. In figure 10.16, the first number in the parentheses is its maximum curvature and the second is its orientation.



(a) θ is the orientation of this token; (b) an example of smooth curve decomposition.

Figure 10.16: Smooth curve decomposition

The similarity between two tokens is measured by the weighted Euclidean distance. The shape similarity is measured according to a non-metric distance. Shape retrieval based on token representation has shown to be robust in the presence of partially occulted objects, translation, scaling and rotation.

4.6 Symbolic representation based on the axis of least inertia

In [17], a method of representing a shape in terms of multi-interval valued type data is proposed. The proposed shape representation scheme extracts symbolic features with reference to the axis of least inertia, which is unique to the shape. The axis of least inertia (ALI) of a shape is defined as the line for which the integral of the square of the distances to points on the shape boundary is a minimum.

Once the ALI is calculated, each point on the shape curve is projected on to ALI. The two farthest projected points say $E1$ and $E2$ on ALI are chosen as the extreme points as shown in figure 10.17. The Euclidean distance between these two extreme points defines the length of ALI. The length of ALI is divided uniformly by a fixed number n; the equidistant points are called feature points. At every feature point chosen, an imaginary line perpendicular to the ALI is drawn. It is interesting to note that these perpendicular lines may intersect the shape curve at several points. The length of each imaginary line in shape region is computed and the col-

lection of these lengths in an ascending order defines the value of the feature at the respective feature point.

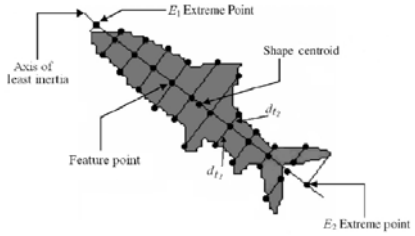


Figure 10.17: Symbolic features based axis of least inertia

Let S be a shape to be represented and n the number of feature points chosen on its ALI. Then the feature vector F representing the shape S, is in general of the form $F=[\mathbf{f}_1,\mathbf{f}_2,...,\mathbf{f}_t,...,\mathbf{f}_n]$, where $f_t=\{d_{t_1},d_{t_2},\cdots,d_{t_k}\}$ for some $t_k\geq 1$.

The feature vector F representing the shape S is then invariant to image transformations viz., uniform scaling, rotation, translation and flipping (reflection).

4.7 Beam angle statistics

Beam angle statistics (BAS) shape descriptor is based on the beams originated from a boundary point, which are defined as lines connecting that point with the rest of the points on the boundary [5].

Let B be the shape boundary. $B=\{P_1,P_2,\cdots,P_N\}$ is represented by a connected sequence of points, $P_i=(x_i, y_i)$, $i=1,2,\cdots,N$, where N is the number of boundary points. For each point P_i, the beam angle between the forward beam vector $V_{i+k}=\overrightarrow{P_iP_{i+k}}$ and backward beam vector $V_{i-k}=\overrightarrow{P_iP_{i-k}}$ in the k^{th} order neighbourhood system, is then computed as (see figure 10.18, $k=5$ for example)

$$C_k(i)=\theta_{V_{i+k}}-\theta_{V_{i-k}} \tag{10.15}$$

where $\theta_{V_{i+k}}=\arctan\frac{y_{i+k}-y_i}{x_{i+k}-x_i}$, $\theta_{V_{i-k}}=\arctan\frac{y_{i-k}-y_i}{x_{i-k}-x_i}$

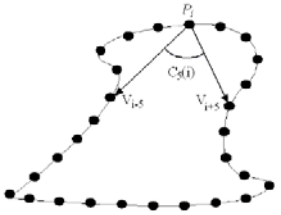
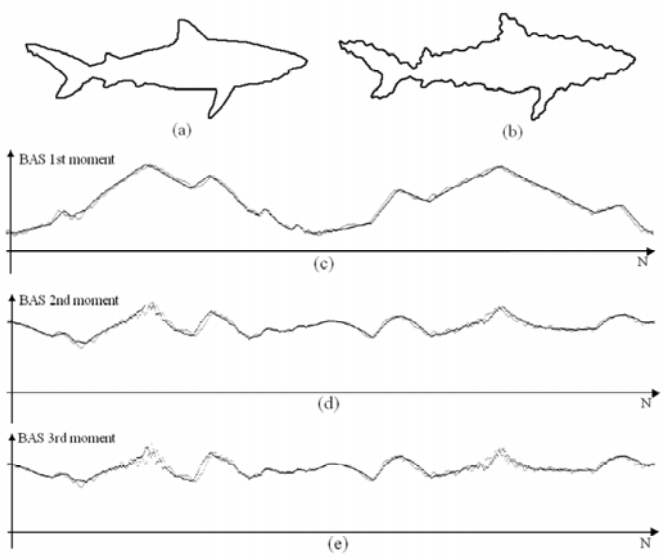


Figure 10.18: Beam angle at the neighbourhood system 5 for a boundary point



(a) Original contour; (b) noisy contour; (c), (d) and (e) are the BAS plot 1st, 2nd and 3rd moment, respectively.

Figure 10.19: The BAS descriptor for original and noisy contour

For each boundary point P_i of the contour, the beam angle $C_k(i)$ can be taken as a random variable with the probability density function $P(C_k(i))$. Therefore, beam angle statistics (BAS), may provide a compact representation for a shape descriptor. For this purpose, m^{th} moment of the random variable $C_k(i)$ is defined as follows:

$$E[(C(i))^m] = \sum_{k=1}^{(N/2)-1} (C_k(i))^m \cdot P_k(C_k(i)), \quad m=1, 2, \cdots \tag{10.16}$$

In the above formula E indicates the expected value. Figure 10.19 shows an example of this descriptor.

Beam angle statistics shape descriptor captures the perceptual information using the statistical information based on the beams of individual points. It gives globally discriminative features to each boundary point by using all other boundary points. BAS descriptor is also quite stable under distortions and is invariant to translation, rotation and scaling.

4.8 Shape matrix

Shape matrix descriptor requires an $M \times N$ matrix to present a region shape. There are two basic modes of shape matrix: Square model [59] and Polar model [44].

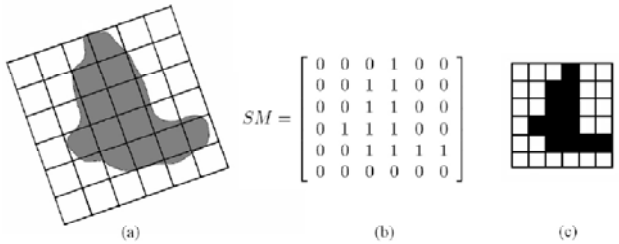
4.8.1 Square model shape matrix

Square model of shape matrix, also called grid descriptor [29, 59], is constructed according to the following algorithm: for the shape S, construct a square centred on the centre of gravity G of S. The size of each side is equal to $2L$, where L is the maximum Euclidean distance from G to a point M on the boundary of the shape. Point M lies in the centre of one side and GM is perpendicular to this side.

Divide the square into $N \times N$ subsquares and denote S_{kj}, $k,j=1,\cdots,N$, the subsquares of the grid. Define the shape matrix $SM=[B_{kj}]$,

$$B_{kj} = \begin{cases} 1 \Leftrightarrow \mu(S_{kj} \bigcap S) \geq \mu(S_{kj})/2 \\ 0 \ \text{otherwise} \end{cases} \tag{10.17}$$

where $\mu(F)$ is the area of the planar region F. Figure 10.20 shows an example of square model of shape matrix.



(a) Original shape region; (b) square model shape matrix; (c) reconstruction of the shape region.

Figure 10.20: Square model shape matrix

For a shape with more than one maximum radius, it can be described by several shape matrices and the similarity distance is the minimum distance between these matrices. In [59], authors gave a method to choose the appropriate shape matrix dimension.

4.8.2 Polar model shape matrix

Polar model of shape matrix is constructed by the following steps. Let G be the centre of gravity of the shape, and GA be the maximum radius of the shape. Using G as centre, draw n circles with radii equally spaced. Starting from GA, and counter clockwise, draw radii that divide each circle into m equal arcs. The values of the matrix are the same as those in square model shape matrix. Figure 10.21 shows an example, where n = 5 and m =12. Its polar model of shape matrix is

$$PSM = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 \\ 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{10.18}$$

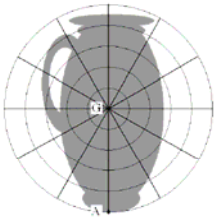


Figure 10.21: Polar model shape

Polar model of shape matrix is simpler than square model because it only uses one matrix no matter how many maximum radii are on the shape. However, since the sampling density is not constant with the polar sampling raster, a weighed shape matrix is necessary. For the detail, refer to [44].

The shape matrix exists for every compact shape. There is no limit to the scope of the shapes that the shape matrix can represent. It can describe even shapes with holes. Shape matrix is also invariant under translation, rotation and scaling of the object. The shape of the object can be reconstructed from the shape matrix; the accuracy is given by the size of the grid cells.

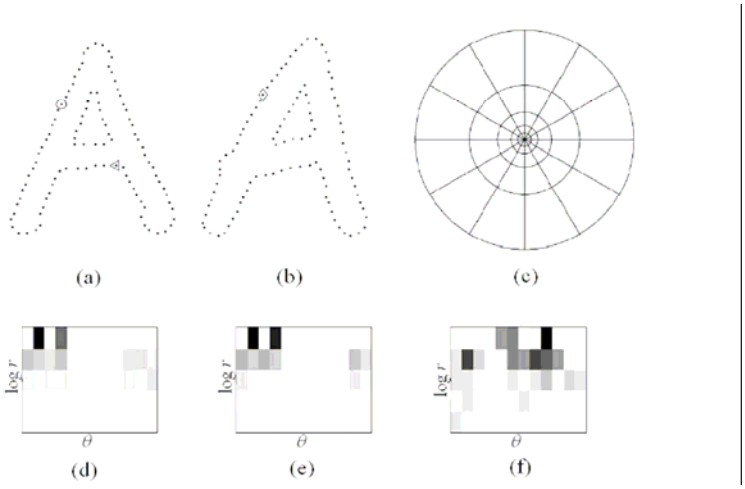
4.9 Shape context

In [8], the shape context has been shown to be a powerful tool for object recognition tasks. It is used to find corresponding features between model and image.

Shape contexts analysis begins by taking N samples from the edge elements on the shape. These points can be on internal or external contours. Consider the vectors originating from a point to all other sample points on the shape. These vectors express the appearance of the entire shape relative to the reference point. This descriptor is the histogram of the relative polar coordinates of all other points:

$$h_i(k) = \#\left\{Q \neq P_i : (Q - P_i) \in bin(k)\right\} \tag{10.19}$$

An example is shown in figure 10.22 where (c) is the diagram of log-polar histogram that has 5 bins for the polar direction and 12 bins for the angular direction. The histogram of a point P_i is formed by the following steps: putting the center of the histogram bins diagram on the point P_i, each bin of this histogram contains a count of all other sample points on the shape falling into that bin. Note that on this figure, the shape contexts (histograms) for the points marked by 'o' (in (a)), '◊' (in (b)) and '◁' (in (a)) are shown in (d), (e) and (f), respectively. It is clear that the shape contexts for the points marked by 'o' and '◊', which are computed for relatively similar points on the two shapes, have visual similarity. By contrast, the shape context for '◁' is quite different from the others. Obviously, this descriptor is a rich description, since as N gets large, the representation of the shape becomes exact.



(a) and (b) Sampled edge points of two shapes; (c) diagram of log-polar histogram bins used in computing the shape contexts; (d), (e) and (f) shape contexts for reference sample points marked by ○, ◇ and ◁ in (a) and (b), respectively. (Dark=large value).

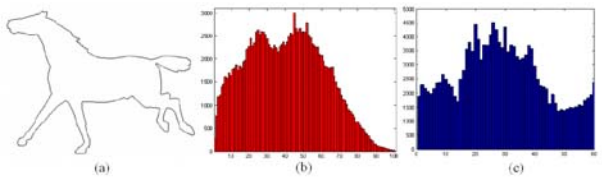
Figure 10.22: Shape context computation and graph matching

Shape context matching is often used to find the corresponding points on two shapes. It has been applied to a variety of object recognition problems [8, 33, 45, 55]. The shape context descriptor has the following invariance properties:

- translation: the shape context descriptor is inherently translation invariant as it is based on relative point locations.
- scaling: for clutter-free images the descriptor can be made scale invariant by normalizing the radial distances by the mean (or median) distance between all point pairs.
- rotation: it can be made rotation invariant by rotating the coordinate system at each point so that the positive x-axis is aligned with the tangent vector.
- shape variation: the shape context is robust against slight shape variations.
- few outliers: points with a final matching cost larger than a threshold value are classified as outliers. Additional 'dummy' points are introduced to decrease the effects of outliers.

4.10 Chord distribution

The basic idea of chord distribution is to calculate the lengths of all chords in the shape (all pair-wise distances between boundary points) and to build a histogram of their lengths and orientations [40]. The "lengths" histogram is invariant to rotation and scales linearly with the size of the object. The "angles" histogram is invariant to object size and shifts relative to object rotation. Figure 10.23 gives an example of chord distribution.



(a) Original contour; (b) chord lengths histogram; (c) chord angles histogram (each stem covers 3)

Figure 10.23: Chord distribution

4.11 Shock graphs

Shock graphs is a descriptor based on the medial axis. The medial axis is the most popular method that has been proposed as a useful shape abstraction tool for the representation and modelling of animate shapes. Skeleton and medial axes have been extensively used for characterizing objects satisfactorily using structures that are composed of line or arc patterns. Medial axis is an image processing operation which reduces input shapes to axial stick-like representations. It is as the loci of centres of bi-tangent circles that fit entirely within the foreground region being considered. Figure 10.24 illustrates the medial axis for a rectangular shape.

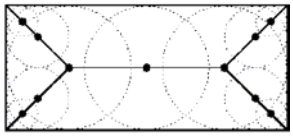


Figure 10.24: Medial axis of a rectangle defined in terms of bi-tangent circles

We notice that the radius of each circle is variable. This variable is a function of the loci of points on the medial axis. We call this function as the radius function.

A shock graph is a shape abstraction that decomposes a shape into a set of hierarchically organized primitive parts. Siddiqi and Kimia define the concept of a shock graph [39] as an abstraction of the medial axis of a shape onto a directed acyclic graph (DAG). Shock segments are curve segments of the medial axis with monotonic flow, and give a more refined partition of the medial axis segments (see figure 10.25).

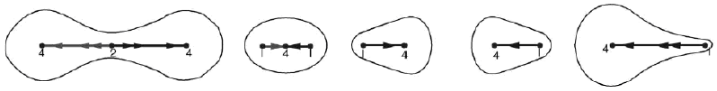


Figure 10.25: Shock segments

The skeleton points are first labelled according to the local variation of the radius function at each point. Shock graph can distinguish the shapes but the medial axis cannot. Figure 10.26 shows two examples of shapes and their shock graphs.

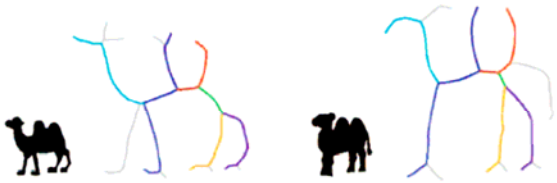


Figure 10.26: Examples of shapes and their shock graphs

To calculate the distance between two shock graphs, in [38], the authors employ a polynomial-time edit-distance algorithm. It shows that this algorithm has good performance against boundary perturbations, articulation and deformation of parts, segmentation errors, scale variations, viewpoint variations and partial occultation.

4.12 Discussions

Spatial feature descriptor is a direct method to describe a shape. These descriptors can apply tree-based theory (Adaptive grid resolution and Convex hull), statistic (Chain code histogram, Beam angle statistics, Shape context and Chord distribution) or syntactic analysis (Smooth curve decomposition) to extract or represent the feature of a shape. This description scheme not only compresses the data of a shape, but also provides a compact and meaningful form to facilitate further recognition operations.

5 Moments

This concept is issued from the concept of moments in mechanics where mass repartition of objects are observed. It is an integrated theory system. For both contour and region of a shape, one can use moment's theory to analyze the object.

5.1 Boundary moments

Boundary moments, analysis of a contour, can be used to reduce the dimension of boundary representation [41]. Assume shape boundary has been represented as a 1-D shape representation $z(i)$ as introduced in Section 2, the r^{th} moment m_r and central moment μ_r can be estimated as

$$m_r = \frac{1}{N}\sum_{i=1}^{N}[z(i)]^r \quad and \quad \mu_r = \frac{1}{N}\sum_{i=1}^{N}[z(i)-m_1]^r \tag{10.20}$$

where N is the number of boundary points.

The normalized moments $\overline{m}_r = m_r/(\mu_2)^{r/2}$ and $\overline{\mu}_r = \mu_r/(\mu_2)^{r/2}$ are invariant to shape translation, rotation and scaling. Less noise-sensitive shape descriptors can

be obtained from

$$F_1=\frac{(\mu_2)^{1/2}}{m_1},\quad F_2=\frac{\mu_3}{(\mu_2)^{3/2}}\quad and\quad F_3=\frac{\mu_4}{(\mu_2)^2} \tag{10.21}$$

The other boundary moments method treats the 1-D shape feature function $z(i)$ as a random variable **v** and creates a K bins histogram $p(v_i)$ from $z(i)$. Then, the r^{th} central moment is obtained by

$$\mu_r=\sum_{i=1}^{K}(v_i-m)^r p(v_i)\quad and\quad m=\sum_{i=1}^{K} v_i p(v_i) \tag{10.22}$$

The advantage of boundary moment descriptors is that they are easy to implement. However, it is difficult to associate higher order moments with physical interpretation.

5.2 Region moments

Among the region-based descriptors, moments are very popular. These include moment invariants, Zernike moments, Radial Chebyshev moments, etc.

The general form of a moment function m_{pq} of order $(p+q)$ of a shape region can be given as:

$$m_{pq}=\sum_x\sum_y \Psi_{pq}(x,y)f(x,y)\qquad p,q=0,1,2\cdots \tag{10.23}$$

where Ψ_{pq} is known as the *moment weighting kernel* or the *basis set*; $f(x,y)$ is the shape region defined as follows

$$f(x,y)=\begin{cases}1 & \text{if } (x,y)\in D\\ 0 & \text{otherwise}\end{cases} \tag{10.24}$$

where D represents the image domain.

5.2.1 Invariant moments (IM)

Invariant moments (IM) are also called geometric moment invariants. Geometric moments, are the simplest of the moment functions with basis $\Psi_{pq}=x^p y^q$, while complete, is not orthogonal [57]. Geometric moment function m_{pq} of order $(p+q)$ is given as:

$$m_{pq}=\sum_x\sum_y x^p y^q f(x,y)\qquad p,q=0,1,2\cdots \tag{10.25}$$

The geometric central moments, which are invariant to translation, are defined as

$$\mu_{pq}=\sum_x\sum_y (x-\bar{x})^p (y-\bar{y})^q f(x,y) \text{ with } p,q=0,1,2\cdots \tag{10.26}$$

where $\bar{x}=m_{10}/m_{00}$ and $\bar{y}=m_{01}/m_{00}$

A set of 7 invariant moments (IM) is given by [57]:

$$\phi_1=\eta_{20}+\eta_{02} \tag{10.27}$$

$$\phi_2=(\eta_{20}-\eta_{02})^2+4\eta_{11}^2 \tag{10.28}$$

$$\phi_3=(\eta_{30}-3\eta_{12})^2+(3\eta_{21}-\eta_{03})^2 \tag{10.29}$$

$$\phi_4=(\eta_{30}+\eta_{12})^2+(\eta_{21}+\eta_{03})^2 \tag{10.30}$$

$$\phi_5=(\eta_{30}-3\eta_{12})(\eta_{30}+\eta_{12})\left[(\eta_{30}+\eta_{12})^2-3(\eta_{21}+\eta_{03})^2\right]+(3\eta_{21}-\eta_{03})(\eta_{21}+\eta_{03}) \cdot\left[3(\eta_{30}+\eta_{12})^2-(\eta_{21}+\eta_{03})^2\right] \tag{10.31}$$

$$\phi_6=(\eta_{20}-\eta_{02})\left[(\eta_{30}+\eta_{12})^2-(\eta_{21}+\eta_{03})^2\right]+4\eta_{11}^2(\eta_{30}+\eta_{12})(\eta_{21}+\eta_{03}) \tag{10.32}$$

$$\phi_7=(3\eta_{21}-\eta_{03})(\eta_{30}+\eta_{12})\left[(\eta_{30}+\eta_{12})^2-3(\eta_{21}+\eta_{03})^2\right]+(3\eta_{12}-\eta_{03})(\eta_{21}+\eta_{03}) \cdot\left[3(\eta_{30}+\eta_{12})^2-(\eta_{21}+\eta_{03})^2\right] \tag{10.33}$$

where $\eta_{pq}=\mu_{pq}/\mu_{00}^{\gamma}$ and $\gamma=1+(p+q)/2$ for $p+q=2,3,\cdots$

IM are computationally simple. Moreover, they are invariant to rotation, scaling and translation. However, they have several drawbacks [10]:

- information redundancy: since the basis is not orthogonal, these moments suffer from a high degree of information redundancy.
- noise sensitivity: higher-order moments are very sensitive to noise.
- large variation in the dynamic range of values: since the basis involves powers of p and q, the moments computed have large variation in the dynamic range of values for different orders. This may cause numerical instability when the image size is large.

5.2.2 Algebraic moment invariants

The algebraic moment invariants are computed from the first m central moments and are given as the eigenvalues of predefined matrices, $M_{[j,k]}$, whose elements are scaled factors of the central moments [43]. The algebraic moment invariants can be constructed up to arbitrary order and are invariant to affine transformations. However, algebraic moment invariants performed either very well or very poorly on the objects with different configuration of outlines.

5.2.3 Zernike moments (ZM)

Zernike Moments (ZM) are orthogonal moments [10]. The complex Zernike moments are derived from orthogonal Zernike polynomials:

$$V_{nm}(x,y)=V_{nm}(r\cos\theta,r\sin\theta)=R_{nm}(r)\exp(jm\theta) \tag{10.34}$$

where $R_{nm}(r)$is the orthogonal radial polynomial:

$$R_{nm}(r)=\sum_{s=0}^{(n-|m|)/2}(-1)^s\frac{(n-s)!}{s!\times\left(\frac{n-2s+|m|}{2}\right)!\left(\frac{n-2s-|m|}{2}\right)!}r^{n-2s} \tag{10.35}$$

n=0,1,2,⋯; $0\le|m|\le n$; and $n-|m|$ is even.

Zernike polynomials are a complete set of complex valued functions that are orthogonal over the unit disk, i.e., $x^2+y^2\le1$. The Zernike moment of order n with repetition m of shape region $f(x,y)$ is given by:

$$Z_{nm}=\frac{n+1}{\pi}\sum_r\sum_\theta f(r\cos\theta,r\sin\theta)\cdot R_{nm}(r)\cdot\exp(jm\theta)\quad r\le1 \tag{10.36}$$

Zernike moments (ZM) have the following advantages [35]:

- rotation invariance: the magnitudes of Zernike moments are invariant to rotation.
- robustness: they are robust to noise and minor variations in shape.
- expressiveness: since the basis is orthogonal, they have minimum information redundancy.

However, the computation of ZM (in general, continuous orthogonal moments) pose several problems:

- coordinate space normalization: the image coordinate space must be transformed to the domain where the orthogonal polynomial is defined (unit circle for the Zernike polynomial).
- numerical approximation of continuous integrals: the continuous integrals must be approximated by discrete summations. This approximation not only leads to numerical errors in the computed moments, but also severely affects the analytical properties such as rotational invariance and orthogonality.
- computational complexity: computational complexity of the radial Zernike polynomial increases as the order becomes large.

5.2.4 Radial Chebyshev moments (RCM)

The radial Chebyshev moment of order p and repetition q is defined as [34]:

$$S_{pq}=\frac{1}{2\pi\rho(p,m)}\sum_{r=0}^{m-1}\sum_{\theta=0}^{2\pi}t_p(r)\cdot\exp(-jq\theta)\cdot f(r,\theta) \tag{10.37}$$

where $t_p(r)$ is the scaled orthogonal Chebyshev polynomials for an $N\times N$ image such that

$$t_p(x)=\frac{(2p-1)t_1(x)t_{p-1}(x)-(p-1)\left\{1-\frac{(p-1)^2}{N^2}\right\}t_{p-2}(x)}{p},\quad p>1 \tag{10.38}$$

with $t_0(x)=1$, $t_1(x)=(2x-N+1)/N$ and where $\rho(p,N)$ is the squared-norm:

$$\rho(p,N)=\frac{N\left(1-\frac{1}{N^2}\right)\left(1-\frac{2^2}{N^2}\right)\cdots\left(1-\frac{p^2}{N^2}\right)}{2p+1},\quad p=0,1,\cdots,N-1 \tag{10.39}$$

and $m=(N/2)+1$.

The mapping between (r,θ) and image coordinates (x,y) is given by:

$$x=\frac{rN}{2(m-1)}\cos(\theta)+\frac{N}{2}\ \text{ and }\ y=\frac{rN}{2(m-1)}\sin(\theta)+\frac{N}{2} \tag{10.40}$$

Compared to Chebyshev moments, radial Chebyshev moments possess rotational invariance property.

5.3 Discussions

Besides the previous moments, there are other moments for shape representation, for example, homocentric polar-radius moment [20], orthogonal Fourier-Mellin moments (OFMMs) [21], pseudo-Zernike Moments [31], etc. The study shows that the moment-based shape descriptors are usually concise, robust and easy to compute. They are also invariant to scaling, rotation and translation of the object. However, because of their global nature, the disadvantage of moment-based methods is that it is difficult to correlate high order moments with a shape's salient features.

6 Scale space approaches

Scale space approaches are issued from multiscale representation that allows handling shape structure at different scales. In scale space theory a curve is embedded into a continuous family $\{\Gamma_\sigma:\sigma\geq 0\}$ of gradually simplified versions. The main idea of scale spaces is that the original curve $\Gamma=\Gamma_0$ should get more and more simplified, and so small structures should vanish as parameter σ increases. Thus due to different scales (values of σ), it is possible to separate small details from relevant shape properties. The ordered sequence $\{\Gamma_\sigma:\sigma\geq 0\}$ is referred to as evolution of Γ.

A lot of shape features can be analyzed in scale-space theory to get more information about shapes. Here we introduced 2 scale-space approaches: curvature scale-space (CSS) and intersection points map (IPM).

6.1 Curvature scale-space

The curvature scale-space (CSS) method, proposed by F. Mokhtarian in 1988, was selected as a contour shape descriptor for MPEG-7. This approach is based on multi-scale representation and curvature to represent planar curves. For convenience, a contour is defined with a discrete parameterization as following:

$$\Gamma(\mu)=(x(\mu),y(\mu)) \tag{10.41}$$

An evolved version of that curve is defined by

$$\Gamma_\sigma(\mu)=(X(\mu,\sigma),Y(\mu,\sigma)) \tag{10.42}$$

where $X(\mu,\sigma)=x(\mu)*g(\mu,\sigma)$ and $Y(\mu,\sigma)=y(\mu)*g(\mu,\sigma)$, * is the convolution operator, and $g(\mu,\sigma)$ denotes a Gaussian filter with standard deviation σ defined by

$$g(\mu,\sigma)=\frac{1}{\sigma\sqrt{2\pi}}\exp\left(\frac{-\mu^2}{2\sigma^2}\right) \tag{10.43}$$

Functions $X(\mu,\sigma)$ and $Y(\mu,\sigma)$ are given explicitly by

$$X(\mu,\sigma)=\int_{-\infty}^{\infty} x(v)\frac{1}{\sigma\sqrt{,2\pi}}\exp\left(\frac{-(\mu-v)^2}{2\sigma^2}\right)dv \tag{10.44}$$

$$Y(\mu,\sigma)=\int_{-\infty}^{\infty} y(v)\frac{1}{\sigma\sqrt{,2\pi}}\exp\left(\frac{-(\mu-v)^2}{2\sigma^2}\right)dv \tag{10.45}$$

The curvature of the contour is given by

$$k(\mu,\sigma)=\frac{X_\mu(\mu,\sigma)Y_{\mu\mu}(\mu,\sigma)-X_{\mu\mu}(\mu,\sigma)Y_\mu(\mu,\sigma)}{(X_\mu(\mu,\sigma)^2-Y_\mu(\mu,\sigma)^2)^{3/2}} \tag{10.46}$$

where

$$X_\mu(\mu,\sigma)=\frac{\partial}{\partial\mu}(x(\mu)*g(\mu,\sigma))=x(\mu)*g_\mu(\mu,\sigma) \tag{10.47}$$

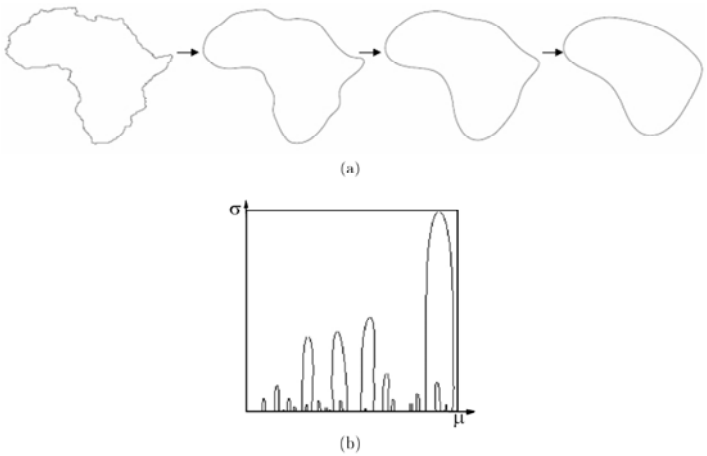
$$X_{\mu\mu}(\mu,\sigma)=\frac{\partial^2}{\partial\mu^2}(x(\mu)*g(\mu,\sigma))=x(\mu)*g_{\mu\mu}(\mu,\sigma) \tag{10.48}$$

$$Y_\mu(\mu,\sigma)=\frac{\partial}{\partial\mu}(y(\mu)*g(\mu,\sigma))=y(\mu)*g_\mu(\mu,\sigma) \tag{10.49}$$

$$Y_{\mu\mu}(\mu,\sigma)=\frac{\partial^2}{\partial\mu^2}(y(\mu)*g(\mu,\sigma))=y(\mu)*g_{\mu\mu}(\mu,\sigma) \tag{10.50}$$

Note that σ is also referred to as a scale parameter. The process of generating evolved versions of Γ_σ as σ increases from 0 to ∞ is referred to as the evolution of Γ_σ. This technique is suitable for removing noise and smoothing a planar curve as well as gradual simplification of a shape.

The function defined by $k(\mu,\sigma)=0$ is the CSS image of Γ. Figure 10.27 is a CSS image examples.



(a) Evolution of Africa: from left to right σ=0(original), σ=4, σ=8 and σ=16, respectively; (b) CSS image of Africa.

Figure 10.27: Curvature scale-space image

The representation of CSS is the maxima of CSS contour of an image. Many methods for representing the maxima of CSS exist in the literatures [19, 36, 52] and the CSS technique has been shown to be robust contour-based shape representation technique. The basic properties of the CSS representation are as follows:

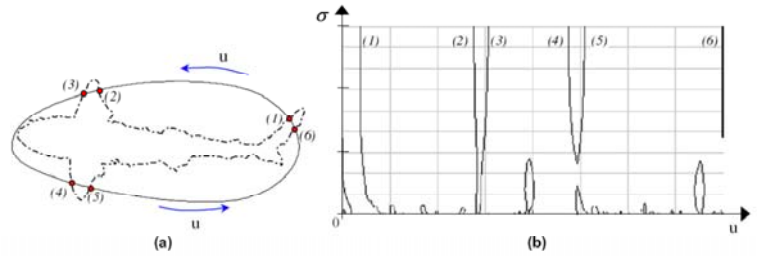
- it captures the main features of a shape, enabling similarity-based retrieval;
- it is robust to noise, changes in scale and orientation of objects;
- it is compact, reliable and fast;
- It retains the local information of a shape. Every concavity or convexity on the shape has its own corresponding contour on the CSS image.

Although CSS has a lot of advantages, it does not always give results in accordance with human vision system. The main drawbacks of this description are due to the problem of shallow concavities/convexities on a shape. It can be shown that the shallow and deep concavities/convexities may create the same large contours on the CSS image. In [1, 49], the authors gave some methods to alleviate these effects.

6.2 Intersection points map

Similarly to the CSS, many methods also use a Gaussian kernel to progressively smooth the curve relatively to the varying bandwidth. In [24], the authors proposed a new algorithm, intersection points map (IPM), based on this principle. Instead of characterizing the curve with its curvature involving 2^{nd} order derivatives, it uses the intersection points between the smoothed curve and the original. As the standard deviation of the Gaussian kernel increases, the number of the intersection

points decreases. By analyzing these remaining points, features for a pattern can be defined. Figure 10.28 represents an example of IPM.



(a) An original contour; (b) an IPM image in the (u,σ) plane. The IPM points indicated by (1)-(6) refer to the corresponding intersection points in (a).

Figure 10.28: Example of the IPM

The IPM pattern can be identified regardless of its orientation, translation and scale change. It is also resistant to noise for a range of noise energy. The main weakness of this approach is that it fails to handle occulted contours and those having undergone a non-rigid deformation. Since this method deals only with curve smoothing, it needs only the convolution operation in the smoothing process. So this method is faster than the CSS one with equivalent performances.

6.3 Discussions

As multi-resolution analysis in signal processing, scale-space theory can obtain abundant information about a contour with different scales. In scale-space, global pattern information can be interpreted from higher scales, while detailed pattern information can be interpreted from lower scales. Scale-space algorithm benefits from the boundary information redundancy in the new image, making it less sensitive to errors in the alignment or contour extraction algorithms. The great advantages are the high robustness to noise and the great coherence with human perception.

7 Shape transform domains

With operators transforming data pixels into frequency domain, a description of shape can be obtained with respect to its frequency content. The transform domain class includes methods which are formed by the transform of the detected object or the transform of the whole image. Transforms can therefore be used to characterize the appearance of images. The shape feature is represented by all or partial coefficients of a transform.

7.1 Fourier descriptors

Although, Fourier descriptor (FD) is a 40-year-old technique, it is still considered

as a valid description tool. The shape description and classification using FD either in contours or regions are simple to compute, robust to noise and compact. It has many applications in different areas.

7.1.1 One-dimensional Fourier descriptors

In general, Fourier descriptor (FD) is obtained by applying Fourier transform on a shape signature that is a one-dimensional function derived from shape boundary coordinates (cf. Section 2). The normalized Fourier transformed coefficients are called the Fourier descriptor of the shape. FD derived from different signatures has significant different performance on shape retrieval. As shown in [52, 53], FD derived from centroid distance function *r(t)* outperforms FD derived from other shape signatures in terms of overall performance. The discrete Fourier transform of *r(t)* is then given by

$$a_n=\frac{1}{N}\sum_{t=0}^{N-1} r(t)\exp\left(\frac{-j2\pi nt}{N}\right), \quad n=0,1,\cdots,N-1 \tag{10.51}$$

Since the centroid distance function *r(t)* is only invariant to rotation and translation, the acquired Fourier coefficients have to be further normalized so that they are scaling and starting point independent shape descriptors. From Fourier transform theory, the general form of the Fourier coefficients of a contour centroid distance function $r(t)$ transformed through scaling and change of start point from the original function $r(t)^{(o)}$ is given by

$$a_n=\exp(jn\tau)\cdot s\cdot a^{(o),n} \tag{10.52}$$

where a_n and $a_n^{(o)}$ are the Fourier coefficients of the transformed shape and the original shape, respectively, τ is the angle incurred by the change of starting point and s is the scale factor. Now considering the following expression:

$$b_n=\frac{a_n}{a_1}=\frac{\exp(jn\tau)\cdot s\cdot a_n^{(o)}}{\exp(j\tau)\cdot s\cdot a_1^{(o)}}=\frac{a_n^{(o)}}{a_1^{(o)}}\exp[j(n-1)\tau]=b_n^{(o)}\exp[j(n-1)\tau] \tag{10.53}$$

where b_n and $b_n^{(o)}$ are the normalized Fourier coefficients of the transformed shape and the original shape, respectively. If we ignore the phase information and only use magnitude of the coefficients, then $|b_n|$ and $\left|b_n^{(o)}\right|$ are the same. In other words, $|b_n|$ is invariant to translation, rotation, scaling and change of start point.

The set of magnitudes of the normalized Fourier coefficients of the shape $\{|b_n|, 0<n<N\}$ are used as shape descriptors, denoted as

$$\{FD_n, 0<n<N\}. \tag{10.54}$$

One-dimensional FD has several interesting characteristics such as simple derivation, simple normalization and simple to do matching. As indicated in [52], for efficient retrieval, 10 FDs are sufficient for shape description.

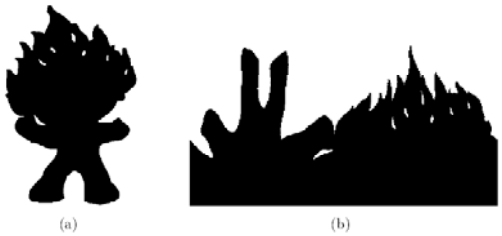
7.1.2 Region-based Fourier descriptor

The region-based FD is referred to as generic FD (GFD), which can be used for general applications. Basically, GFD is derived by applying a modified polar Fourier transform (MPFT) on shape image [48, 54]. In order to apply MPFT, the polar shape image is treated as a normal rectangular image. The steps are as follows

1. the approximated normalized image is rotated counter clockwise by an angular step sufficiently small.
2. the pixel values along positive x-direction starting from the image center are copied and pasted into a new matrix as row elements.
3. the steps 1 and 2 are repeated until the image is rotated by 360°.

The result of these steps is that an image in polar space plots into Cartesian space.

Figure 10.29 shows the polar shape image turning into normal rectangular image.



(a) Original shape image in polar space; (b) polar image of (a) plotted into Cartesian space.

Figure 10.29: The polar shape image turns into normal rectangular image.

The Fourier transform is obtained by applying a discrete 2D Fourier transform on this shape image, so that

$$pf(\rho,\phi)=\sum_{r}\sum_{i} f(r,\theta_i)exp[j2\pi(\frac{r}{R}\rho+\frac{2\pi i}{T}\phi)] \tag{10.55}$$

where $0\le r=\sqrt{[(x-g_x)^2+(y-g_y)^2]}<R$, and $\theta_i=i(2\pi/T)$; $0\le\rho<R$, $0\le\phi<T$ with (g_x,g_y) being the centre of mass of the shape; R and T are the radial and angular resolutions. The acquired Fourier coefficients are translation invariant. Rotation and scaling invariance are achieved by the following:

$$GFD=\left\{\frac{|pf(0,0)|}{area},\frac{|pf(0,1)|}{|pf(0,0)|},\ldots,\frac{|pf(0,n)|}{|pf(0,0)|},\ldots,\frac{|pf(m,0)|}{|pf(0,0)|},\ldots,\frac{|pf(m,n)|}{|pf(0,0)|}\right\} \tag{10.56}$$

where *area* is the area of the bounding circle in which the polar image resides. m is the maximum number of the radial frequencies selected and n is the maximum number of selected angular frequencies. m and n can be adjusted to achieve hierarchical coarse to fine representation requirement.

For efficient shape description, following the implementation of [54], 36 GFD features reflecting m=4 and n=9 are selected to index the shape. The experimental results have shown GFD as invariant to translation, rotation, and scaling. For obtaining the affine and general minor distortions invariance, in [54], the authors

proposed Enhanced Generic Fourier Descriptor (EGFD) to improve the GFD properties.

7.2 Wavelet transform

A hierarchical planar curve descriptor is developed by using the wavelet transform [13]. This descriptor decomposes a curve into components of different scales so that the coarsest scale components carry the global approximation information while the finer scale components contain the local detailed information. The wavelet descriptor has many desirable properties such as multi-resolution representation, invariance, uniqueness, stability, and spatial localization. In [23], the authors use dyadic wavelet transform deriving an affine invariant function. In [12], a descriptor is obtained by applying the Fourier transform along the axis of polar angle and the wavelet transform along the axis of radius. This feature is also invariant to translation, rotation, and scaling. At same time, the matching process of wavelet descriptor can be accomplished cheaply.

7.3 Angular radial transformation

The angular radial transformation (ART) is based in a polar coordinate system where the sinusoidal basis functions are defined on a unit disc. Given an image function in polar coordinates, $f(\rho,\theta)$, an ART coefficient F_{nm} (radial order n, angular order m) can be defined as [37]:

$$F_{nm}=\int_0^{2\pi}\int_0^{1} V_{nm}(\rho,\theta)f(\rho,\theta)\rho d\rho d\theta \tag{10.57}$$

where $V_{nm}(\rho,\theta)$ is the ART basis function and is separable in the angular and radial directions so that:

$$V_{nm}(\rho,\theta)=A_m(\theta)R_n(\rho) \tag{10.58}$$

The angular basis function, A_m, is an exponential function used to obtain orientation invariance. This function is defined as:

$$A_m(\theta)=\frac{1}{2\pi}e^{jm\theta} \tag{10.59}$$

where R_n, the radial basis function, is defined as:

$$R_n(\rho)=\begin{cases}1 & \text{if } n=0\\ 2\cos(\pi n\rho) & \text{if } n\neq 0\end{cases} \tag{10.60}$$

In MPEG-7, twelve angular and three radial functions are used (n<3,m<12). Real parts of the 2-D basis functions are shown in figure 10.30.

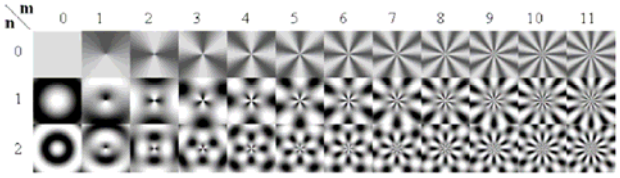


Figure 10.30: Real parts of the ART basis functions

For scale normalization, the ART coefficients are divided by the magnitude of ART coefficient of order n=0,m=0. MPEG-7 standardization process showed the efficiency of 2-D angular radial transformation. This descriptor is robust against translation, scaling, multi-representation (remeshing, weak distortions) and noises.

7.4 Shape signature harmonic embedding

A harmonic function is obtained by a convolution between the Poisson kernel $P_R(r,\theta)$ and a given boundary function $u(Re^{j\phi})$. Poisson kernel is defined by

$$P_R(r,\theta)=\frac{R^2-r^2}{R^2-2Rr\cos(\theta)+r^2} \tag{10.61}$$

The boundary function could be any real- or complex-valued function, but here we choose shape signature functions for the purpose of shape representation. For any shape signature $s[n]$,n=0,1,⋯,N–1, the boundary values for a unit disk can be set as

$$u(Re^{j\phi})=u(Re^{j\omega_0 n})=s[n] \tag{10.62}$$

where $\omega_0=2\pi/N$, $\phi=\omega_0 n$.

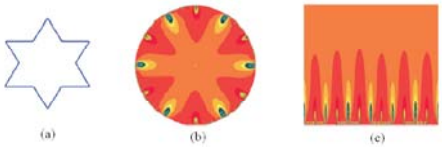
So the harmonic function u can be written as

$$u(re^{j\theta})=\frac{1}{2\pi}\int_0^{2\pi} u(Re^{j\phi})P_R(r,\phi-\theta)d\phi \tag{10.63}$$

The Poisson kernel $P_R(r,\theta)$ has a low-pass filter characteristic, where the radius r is inversely related to the bandwidth of the filter. The radius r is considered as the scale parameter of a multi-scale representation [27]. Another important property is $P_R(0,\theta)=1$, indicating $u(0)$ is the mean value of boundary function $u(Re^{j\phi})$.

In [27], the authors proposed a formulation of a discrete closed-form solution for the Poisson's integral formula of equation (10.63), so that one can avoid the need for approximation or numerical calculation of the Poisson summation form.

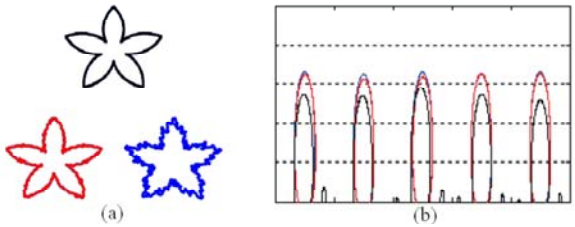
As in Subsection 7.1.2, the harmonic function inside the disk can be mapped to a rectilinear space for a better illustration. Figure 10.31 shows an example for a star shape. Here, we used curvature as the signature to provide boundary values.



(a) Example shape; (b) harmonic function within the unit disk; (c) rectilinear mapping of the function.

Figure 10.31: Harmonic embedding of curvature signature

The zero-crossing image of the harmonic functions is extracted as a shape feature. This shape descriptor is invariant to translation, rotation and scaling. It is also robust to noise. Figure 10.32 is an example. The original curve is corrupted with different noise levels, and the harmonic embeddings show robustness to the noise.



(a) Original and noisy shapes; (b) harmonic embedding images for centroid distance signature.

Figure 10.32: Centroid distance signature harmonic embedding that is robust to noisy boundaries

In addition, it is more efficient than CSS descriptor. However, it is not suitable for similarity retrieval, because it is inconsistent with non-rigid transform.

7.5 ℜ-Transform

The ℜ-Transform to represent a shape is based on the Radon transform. The approach is presented as follows. We assume that the function f is the domain of a shape. Its Radon transform is defined by:

$$T_R(\rho,\theta)=\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(x,y)\delta(x\cos\theta+y\sin\theta-\rho)dxdy \tag{10.64}$$

where $\delta(.)$ is the Dirac delta-function such that:

$$\delta(x)=\begin{cases}1 & \text{if } x=0\\ 0 & \text{otherwise}\end{cases} \tag{10.65}$$

$\theta\in[0,\pi]$ and $\rho\in(-\infty,\infty)$. In other words, Radon transform $T_R(\rho,\theta)$ is the integral of f over the line $L_{(\rho,\theta)}$ defined by $\rho=x\cos\theta+y\sin\theta$.

Figure 10.33 is an example of a shape and its Radon transform.

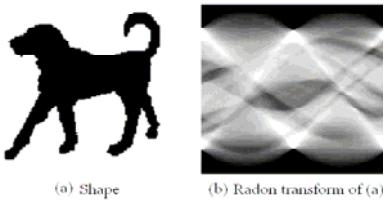


Figure 10.33: A shape and its Radon transform

The following transform is defined as $\mathcal{R}$-transform:

$$\mathcal{R}_f(\theta) = \int_{-\infty}^{-\infty} T_R^2(\rho,\theta)d\rho \tag{10.66}$$

where $T_R(\rho,\theta)$ is the Radon transform of the domain function *f*. In [42], the authors show the following properties of $\mathcal{R}_f(\theta)$:

- periodicity: $\mathcal{R}_f(\theta\pm\pi)=\mathcal{R}_f(\theta)$
- rotation: a rotation of the image by an angle θ_0 implies a translation of the $\Re$-transform of θ_0: $\mathcal{R}_f(\theta+\theta_0)$.
- translation: the $\Re$-transform is invariant under a translation of the shape *f* by a vector $\vec{u} = (x_0, y_0)$.
- scaling: a change of the scaling of the shape *f* induces a scaling only in the amplitude of the $\mathcal{R}$-transform.

Given a large collection of shapes, one $\mathcal{R}$-transform per shape is not efficient to distinguish from the others because the $\mathcal{R}$-transform provides a highly compact shape representation. In this perspective, to improve the description, each shape is projected in the Radon space for different segmentation levels of the Chamfer distance transform. Chamfer distance transform is introduced in [60, 61].

Given the distance transform of a shape, the distance image is segmented into *N* equidistant levels to keep the segmentation isotropic. For each distance level, pixels having a distance value superior to that level are selected and at each level of segmentation, an $\mathcal{R}$-transform is computed. In this manner, both the internal structure and the boundaries of the shape are captured. Since a rotation of the shape implies a corresponding shift of the $\mathcal{R}$-transform. Therefore, a one-dimensional Fourier transform is applied on this function to obtain the rotation invariance. After the one-dimensional discrete Fourier transform *F*, $\mathcal{R}$-transform descriptor vector is defined as follows:

$$RTD=\left(\frac{F\mathcal{R}^1(\frac{\pi}{M})}{F\mathcal{R}^1(0)},\cdots,\frac{F\mathcal{R}^1(\frac{i\pi}{M})}{F\mathcal{R}^1(0)},\cdots,\frac{F\mathcal{R}^1(\pi)}{F\mathcal{R}^1(0)},\cdots,\frac{F\mathcal{R}^N(\frac{\pi}{M})}{F\mathcal{R}^N(0)},\cdots,\frac{F\mathcal{R}^N(\frac{i\pi}{M})}{F\mathcal{R}^N(0)},\cdots,\frac{F\mathcal{R}^N(\pi)}{F\mathcal{R}^N(0)}\right) \tag{10.67}$$

where $i\in[1,M]$, *M* is the angular resolution, $F\mathcal{R}^\alpha$ is the magnitude of Fourier transform to $\mathcal{R}$-transform and $\alpha\in[1,N]$, is the segmentation level of Chamfer dis-

tance transform.

7.6 Shapelet descriptor

Shapelet descriptor was proposed to present a model for animate shapes and for extracting meaningful parts of objects. The model assumes that animate shapes (2D simple closed curves) are formed by a linear superposition of a number of shape bases. A basis function $\psi(s;\mu,\sigma)$ is defined in [15] so that $\mu\in[0,1]$ indicates the location of the basis function relative to the domain of the observed curve, and σ is the scale of the function ψ. Figure 10.34 shows the shape of the basis function ψ at different σ values. It displays variety with different parameter and transforms.

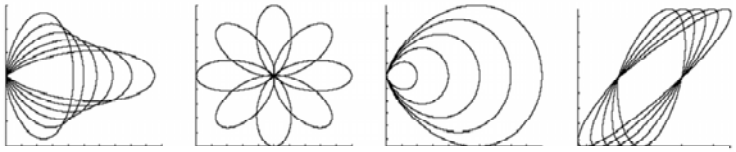


Figure 10.34: Each shape base is a lobe-shaped curve

The basis functions are subject to affine transformations by a 2×2 matrix of basis coefficients:

$$A_k=\begin{bmatrix} a_k & b_k \\ c_k & d_k \end{bmatrix} \tag{10.68}$$

The variables for describing a base are denoted by $\mathbf{b}_k=(A_k,\mu_k,\sigma_k)$ and are termed basis elements. The shapelet is defined by

$$\gamma(s;\mathbf{b}_k)=A_k\psi(s;\mu_k,\sigma_k) \tag{10.69}$$

Figure 10.34(b,c,d) demonstrates shapelets obtained from the basis functions ψ by the affine transformations of rotation, scaling, and shearing respectively, as indicated by the basis coefficient A_k. By collecting all the shapelets at various μ, σ, A and discretizing them at multiple levels, a dictionary is obtained

$$\Delta=\left\{\gamma(s;b):\forall b;a\gamma_0,a>0\right\}. \tag{10.70}$$

A special shapelet γ_0 is defined as an ellipse. Shapelets are the building blocks for shape contours, and they form closed curves by linear addition:

$$\Gamma(s)=\begin{bmatrix} x_0 \\ y_0 \end{bmatrix}+\sum_{k=1}^{K}\begin{bmatrix} a_kb_k \\ c_kd_k \end{bmatrix}\psi(s;\mu_k,\sigma_k)+\mathbf{n}(s) \tag{10.71}$$

where (x_0,y_0) is the centroid of the contour and **n** is residue.

A discrete representation $\mathbf{B}=(K,\mathbf{b}_1,\mathbf{b}_2,\cdots,\mathbf{b}_K)$, shown by the dots in second row of figure 10.35, represents a shape. **B** is called the "shape script" by analogy to music scripts, where each shapelet is represented by a dot in the (μ,σ) domain. The horizontal axis is $\mu\in[0,1]$ and the vertical axis is the σ. Large dots correspond to big coefficient matrix

$$A_k^2=a_k^2+b_k^2+c_k^2+d_k^2 \tag{10.72}$$

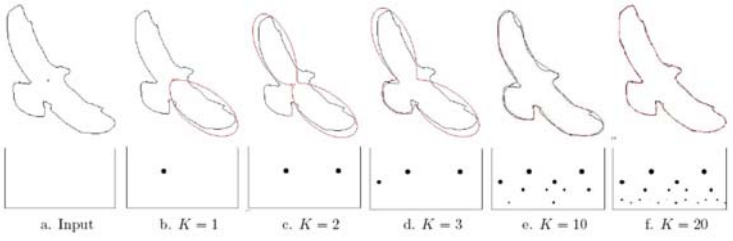


Figure 10.35: Pursuit of shape bases for an eagle contour

Clearly, computing the shape script **B** is a non-trivial task, since Δ is over-complete and there will be multiple sets of bases that reconstruct the curve with equal precision. [15] gave some pursuit algorithms to use shapelets representing a shape.

7.7 Discussions

As a kind of global shape description technique, shape analysis in transform domains takes the whole shape as the shape representation. The description scheme is designed for this representation. Unlike the spatial interrelation feature analysis, shape transform projects a shape contour or region into an other domain to obtain some of its intrinsic features. For shape description, there is always a trade-off between accuracy and efficiency. On one hand, shape should be described as accurate as possible; on the other hand, shape description should be as compact as possible to simplify indexing and retrieval. For a shape transform analysis algorithm, it is very flexible to accomplish a shape description with different accuracy and efficiency by choosing the number of transform coefficients.

8 Summary table

For convenience, to compare these shape feature extraction approaches in this chapter, we summarize their properties in Table 10.1.

Frankly speaking, it is not equitable to affirm a property of an approach by rudely speaking “good” or “bad” because certain approaches have great differences in performances under different conditions.

For example, the method area function is invariant with affine transform under the condition of the contours sampled at its same vertices; whereas it is not robust to affine transform if the condition can’t be contented. In addition, some approaches have good properties for certain type shapes; however it is not for the others. For example, the method shapelets representation is especially suitable for blobby objects, and it has shortcomings in representing elongated objects.

Table 10.1: Properties of shape feature extraction approaches

Shape representation			Recon-struc-ture	Invariance				Resistance			Computational complexity
				Translation	Rotation	Scale	Affine transform	Noise	Occultation	Non-rigid deformation	
Shape signatures	Complex coordinates		Yes	Bad	Bad	Bad	Bad	Average	Good	Bad	Low
	Central distance		No	Good	Good	Good	Bad	Average	Good	Bad	Low
	Tangent angle		No	Good	Good	Good	Bad	Bad	Good	Average	Low
	Curvature function		No	Good	Good	Good	Bad	Bad	Good	Average	Low
	Area function		No	Good	Good	Good	Good	Good	Good	Bad	Low
	Triangle-area representation		No	Good	Good	Good	Good	Good	Average	Average	Low
	Chord length function		No	Good	Good	Good	Bad	Bad	Bad	Bad	Low
Polygonal approximation	Merging	Distance threshold	No	Good	Good	Good	Bad	Good	Bad	Bad	Average
		Tunneling	No	Good	Good	Good	Bad	Good	Bad	Bad	Average
		Polygon evolution	No	Good	Good	Good	Bad	Good	Bad	Bad	Average
	Splitting		No	Good	Good	Good	Bad	Good	Bad	Bad	Average
Space interrelation Feature	Adaptive grid resolution		Yes	Good	Good	Good	Bad	Good	Good	Bad	Low
	Bounding box		Yes	Good	Good	Good	Average	Good	Good	Average	Average
	Convex hull		No	Good	Good	Good	Good	Average	Bad	Bad	High
	Chain code	Basic chain code	Yes	Good	Bad	Bad	Bad	Bad	Good	Bad	Low
		Vertex chain code	Yes	Good	Bad	Bad	Bad	Bad	Good	Bad	Low
		Statistic chain code	No	Good	Bad	Bad	Bad	Bad	Bad	Bad	Low
	Smooth curve decomposition		No	Good	Good	Good	Bad	Good	Good	Average	Average
	ALI-based representation		No	Good	Good	Good	Average	Good	Average	Bad	Average
	Beam angle statistics		No	Good	Good	Good	Bad	Good	Bad	Bad	Low
	Shape matrix	Square model	Yes	Good	Good	Good	Bad	Bad	Good	Bad	Average
		Polar model	Yes	Good	Good	Good	Bad	Bad	Good	Bad	Low
	Shape context		No	Good	Good	Good	Bad	Bad	Average	Average	Average
	Chord distribution		No	Good	Good	Good	Bad	Good	Bad	Bad	Low
	Shock graphs		Yes	Good	Good	Good	Good	Good	Good	Good	High
Moments	Boundary moments		No	Good	Good	Good	Bad	Average	Bad	Bad	Low
	Region moments	Invariant moments	No	Good	Good	Good	Bad	Bad	Bad	Bad	Average
		Algebraic Moment	No	Good	Good	Good	Good	Average	Bad	Bad	Average
		Zernike Moments	No	Good	Good	Good	Bad	Good	Average	Average	High
		Radial Chebyshev Moments	No	Good	Good	Good	Bad	Good	Average	Average	High
Scale-space methods	Curvature scale space		No	Good	Good	Good	Average	Good	Good	Average	Average
	Intersection points map		No	Good	Good	Good	Average	Good	Good	Bad	Average
Shape transform domains	Fourier descriptors	1-D Fourier descriptor	No	Good	Good	Good	Bad	Bad	Bad	Bad	Average
		Region-based Fourier descriptor	No	Good	Good	Good	Good	Good	Average	Average	High
	Wavelet transform		No	Good	Good	Good	Good	Average	Average	Bad	Average
	Angular radial transformation		No	Good	Good	Good	Bad	Good	Bad	Bad	High
	Signature harmonic embedding		No	Good	Good	Good	Average	Good	Average	Bad	High
	$\Re$ -Transform		No	Good	Good	Good	Bad	Good	Average	Average	High
	Shapelets descriptor		No	Good	Good	Good	Bad	Good	Bad	Bad	High

So the simple evaluations in this table are only as a reference. These evaluations are drawn by assuming that all the necessary conditions have been contented for each approach.

9 Illustrative example: a contour-based shape descriptor

In this section is presented a new contour-based shape descriptor we are developing: it belongs to the class of scale-space methods. Fundamental concepts about affine transforms are introduced, the method and its properties are presented and the method is then evaluated by applying it to shape retrieval from the MPEG-7 CE-Shape-1 database that consists of 1 400 contours.

9.1 Fundamental Concepts

Thereafter fundamental concepts are introduced and defined: the affine transform and 2 parameters which are linear (affine invariant) under affine transforms.

9.1.1 Closed curve

Let us consider the discrete parametric equation of a closed curve Γ:

$$\Gamma(\mu) = (x(\mu), y(\mu)) \tag{10.73}$$

where $\mu \in \{0, \ldots, N-1\}$; an application curve may be parameterized with any number of vertices *N*.

9.1.2 Affine transforms

The affine transformed version of a shape can be represented by the following equations:

$$\begin{bmatrix} x_a(\mu) \\ y_a(\mu) \end{bmatrix} = \begin{bmatrix} a & b \\ c & d \end{bmatrix} \begin{bmatrix} x(\mu) \\ y(\mu) \end{bmatrix} + \begin{bmatrix} e \\ f \end{bmatrix} = A \begin{bmatrix} x(\mu) \\ y(\mu) \end{bmatrix} + B \tag{10.74}$$

where $x_a(\mu)$ and $y_a(\mu)$ represent the coordinates of the transformed shape. Translation is represented by matrix B, while scaling, rotation and shear are reflected in matrix A. Corresponding values of coefficients of *A* can be found in the following matrices:

$$A_{Scaling} = \begin{bmatrix} S_x & 0 \\ 0 & S_y \end{bmatrix}, A_{Rotation} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}, A_{Shear} = \begin{bmatrix} 1 & k \\ 0 & 1 \end{bmatrix} \tag{10.75}$$

If S_x is equal to S_y, $A_{Scaling}$ represents uniform scaling and shape is not deformed under rotation, uniform scaling and translation. However, non-uniform scaling and shear contribute to shape deformation under general affine transforms.

9.1.3 Affine invariant parameters

The *arc length* parameter observed on a closed contour transforms linearly under any linear transformation up to the similarity transform. Translation and rotation do not affect the *arc length*; scaling scales the parameter by the same amount. An arbitrary choice of a starting point only introduces a shift in the parameter. However, the *arc length* is nonlinearly transformed under an affine transform and would not be a suitable parameter in this situation [46].

There are two parameters which are linear under affine transforms. They are the *affine arc length*, and the *enclosed area*.

The first parameter can be derived from the properties of determinants. It is defined as follows:

$$\tau = \int_{\alpha}^{\beta} [x'(s)y''(s) - x''(s)y'(s)]^{1/3} ds \tag{10.76}$$

where *x(s)* and *y(s)* are the coordinates of points on the contour and α and β are the curvilinear abscissa of 2 points on it.

The second affine invariant parameter is *enclosed area*, which is based on the property of affine transforms: under affine mapping, all areas are changed in the same ratio. Based on this property, Arbter et al.[4] defined a parameter ψ, which is linear under a general affine transform, so that:

$$\psi = \frac{1}{2} \int_{\alpha}^{\beta} |x(s)y'(s) - y(s)x'(s)| ds \tag{10.77}$$

where *x(s)* and *y(s)* are the coordinates of points on the contour with the origin of the system located at the centroid of the contour and α and β the curvilinear abscissa of 2 points on it. The parameter ψ is essentially the cumulative sum of triangular areas produced by connecting the centroid to pairs of successive vertices on the contour.

9.2 Equal area normalization

All points on a contour could be expressed in terms of the parameter of index points along the contour curve from a specified starting point. With affine transforms, the position of each point changes and it is possible that the number of points between two specified points changes too. So if we parameterize the contour using the equidistant vertices, the index point along the contour curve will change under affine transforms. For example, figure 10.36(a) is the top view of a plane, and (e) is its rear top view, so (e) is one of possible affine transforms of image (a). Via region segmentation or edge following, we obtain the contours of the two planes (b) and (f). (c) and (g) are parts of the contours (b) and (f) normalized by equidistant vertices respectively. In figure 10.36(c), the number of points on the segment between the points A and B is 21; however, the number is 14 in the same segment in figure 10.36(g). So the contour normalised by equidistant vertices is variant under possible affine transforms.

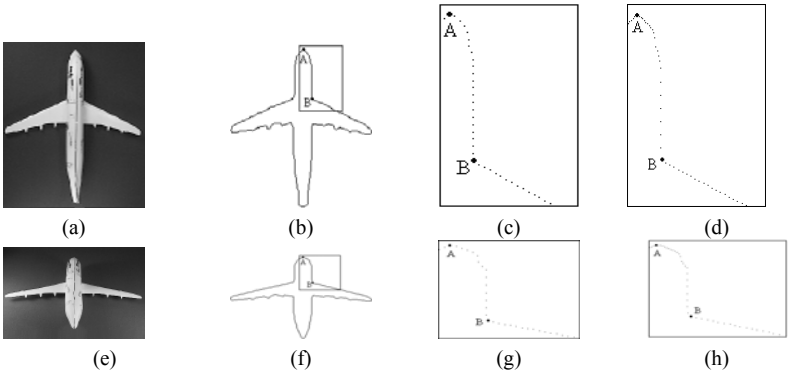


Figure 10.36: The comparison of equidistant vertices normalization and equal area normalization. (a) is the image of the top view of a plane. (b) is the contour of image (a). (c) is a part of contour (b) normalized by equidistant vertices. (d) is a part of contour (b) normalized by equal area. (e) is the image of rear top view of the plane. (f) is the contour of image (e). (g) is a part of contour (f) normalized by equidistant vertices. (h) is a part of contour (f) normalized by equal area.

In order to make it be invariant under affine transforms, a novel curve normalization approach is proposed, which provides an affine invariant description of object curves at low computational cost, while at the same time preserving all information on curve shapes. We call this approach "equal area normalization" (EAN).

All points on a shape contour could be expressed in terms of two functions $\hat{\Gamma}(m) = (\hat{x}(m), \hat{y}(m))$, $m \in [0, M-1]$, where variable m is measured along the contour curve from a specified starting point and M is the total number of points on the contour. The steps of EAN are presented as follows:

1) Normalize $\hat{\Gamma}(m)$ to N points with equidistant vertices. The new functions are denoted $\overline{\Gamma}(\mu) = (\overline{x}(\mu), \overline{y}(\mu))$ and all the points on the contour are $\overline{P}_\mu$, where $\mu \in [0, N-1]$.
2) Calculate the second-order moments of the contour at its centroid G.
3) Transfer the contour to make its centroid G be the origin of the system.
4) Point $\overline{P}_N(\overline{x}(N), \overline{y}(N))$ is assumed to be the same as the first point $\overline{P}_0(\overline{x}(0), \overline{y}(0))$. Compute the area of the contour using the formula:

$$S = \frac{1}{2}\sum_{\mu=0}^{N-1} \left| \overline{x}(\mu)\overline{y}(\mu+1) - \overline{x}(\mu+1)\overline{y}(\mu) \right| \tag{10.78}$$

where $\frac{1}{2}\left|\overline{x}(\mu)\overline{y}(\mu+1) - \overline{x}(\mu+1)\overline{y}(\mu)\right|$ is the area of the triangle whose vertices are $\overline{P}_\mu(\overline{x}(\mu), \overline{y}(\mu))$, $\overline{P}_{\mu+1}(\overline{x}(\mu+1), \overline{y}(\mu+1))$, and centroid G (see figure 10.37).

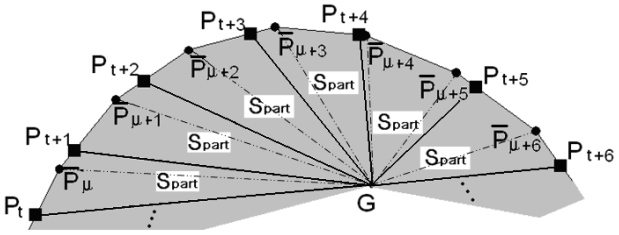


Figure 10.37: The method of equal area normalization. "●" is the vertex $\overline{P}$ of equidistant vertices normalization, and "■" is the point P of equal area normalization. G is the centroid of the contour.

5) Let the number of points on the contour after EAN be N. Of course, any other number of points could be chosen. Therefore, after EAN, each enclosed area S_{part} defined by any two successive points on the contour and the centroid G is equal to $S_{part} = S / N$.

6) Suppose all the points on the contour after EAN are P_t . Let $\Gamma(t) = (x(t), y(t))$ represent the contour, where $t \in [0, N-1]$. Select point $\overline{P}_0(\bar{x}(0), \bar{y}(0))$ on the equidistant vertices normalization as the starting point P_0 $(x(0), y(0))$ of the EAN. On segment $P_0\overline{P}_1$, we seek a point $P_1(x(1), y(1))$, so that the area $S(0)$ of the triangle whose vertices are P_0 $(x(0), y(0))$, $P_1(x(1), y(1))$ and $G(0,0)$ is equal to S_{part} . If there is no point to satisfy this condition, we seek the point P_1 on the segment $\overline{P}_1\overline{P}_2$. So area $S(0)$, which is the sum of the areas of triangle $\overline{P}_0\overline{P}_1G$ and triangle $\overline{P}_1P_1G$, is equal to S_{part} . If again there is yet no point to satisfy the condition, we continue to seek for the point in the next segment until the condition is satisfied. This point P_1 is the second point on the normalized contour.

7) From point $P_1(x(1), y(1))$, we use the same method to calculate all the other points $P_t(x(t), y(t))$, $t \in [2, N-1]$ along the contour. Because the area of each closed zone, e.g. the polygon $P_t[\overline{P}_\mu\overline{P}_{\mu+1}\cdots]P_{t+1}G$ where $t \in [0, N-2]$ is equal to S_{part} , the total area of $N-1$, polygon is equal to $(N-1)\cdot S_{part}$. According to the step 5, the area of the last zone $\overline{P}_{N-1}[\overline{P}_\mu\overline{P}_{\mu+1}\cdots\overline{P}_{N-1}]P_0G$ is exactly equal to:

$$S-(N-1)\cdot S_{part} = N\cdot S_{part} - (N-1)\cdot S_{part} = S_{part}$$

From figure 10.37 we know that the area of triangle $P_tP_{t+1}G$ is approximately equal to the area S_{part} of polygon $P_t[\overline{P}_\mu\overline{P}_{\mu+1}\cdots]P_{t+1}G$ if the two points $\overline{P}_\mu$ and $\overline{P}_{\mu+1}$ are close enough or the number N of the points on the contour is large enough. Therefore, we can use the points P_t , $t \in [0, N-1]$ to replace the points $\overline{P}_\mu$,

$\mu \in [0, N-1]$; the EAN process is then complete.

After this normalization, the number of vertices on the segment between the two appointed points is invariant under affine transforms. Figure 10.36(d) and (h) are the same parts of figure 10.36(c) and (g) respectively. We notice that the distance between the consecutive points is not uniform. In figure 10.36(d), the number of points between points A and B is 23, the number is also 23 in figure 10.36(g). Therefore, after applying EAN, the index of the points on a contour can remain stable with their positions under affine transforms. This property will be very advantageous when extracting the robust attributes of a contour and decreasing complexity in the measurement of similarity. We can also use EAN with the other algorithms, to improve their robustness with affine transforms. For example, before applying the curvature scale space (CSS) algorithm [32], the contour can be normalized by EAN: none of the maximum points in the CSS image will change under affine transforms. This is beneficial when calculating the similarity between two CSS attributes.

9.3 Normalized part area vector

In this section, we look for the existing relations between the part area S_{part}, affine transforms and low-pass filtering.

THEOREM1:

Let $\Gamma_a(\mu)=(x_a(\mu), y_a(\mu))$ be the transformed version of a curve $\Gamma(\mu)= (x(\mu), y(\mu))$ under an affine transform A, where μ is an arbitrary parameter, $\Gamma_{af}(\mu)=(x_{af}(\mu), y_{af}(\mu))$ notes that $\Gamma_a(\mu)$ is filtered by a linear low-pass filter F. Let $\Gamma_f(\mu)=(x_f(\mu), y_f(\mu))$ note that $\Gamma(\mu)$ is filtered by the same low-pass filter F, and $\Gamma_{fa}(\mu)=(x_{fa}(\mu), y_{fa}(\mu))$ refers to the transformed version of $\Gamma_f(\mu)$ under the same affine transform A. The curve $\Gamma_{af}(\mu)$ is then the same as curve $\Gamma_{fa}(\mu)$. In other words: $F(A(\Gamma(\mu)))=A(F(\Gamma(\mu)))$. The following figure illustrates this theorem.

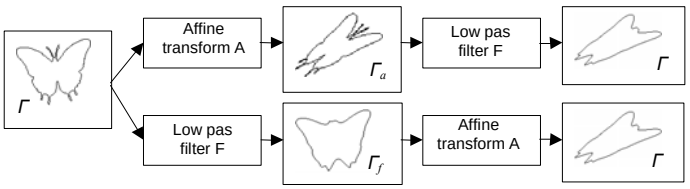


Figure 10.38: Illustration of theorem1

PROOF:

From (10.74) we have

$$x_a(\mu) = ax(\mu) + by(\mu) + e \tag{10.79}$$

$$y_a(\mu) = cx(\mu) + dy(\mu) + f \tag{10.80}$$

For the entire contour, we transfer its centre of gravity to the origin of the system, so that the translation e and f can be removed. Therefore, the affine transform can be represented by two simple formulae:

$$x_a(\mu) = ax(\mu) + by(\mu) \tag{10.81}$$

$$y_a(\mu) = cx(\mu) + dy(\mu) \tag{10.82}$$

The computation starts by convolving each coordinate of the curve $\Gamma_a(\mu)$ with a linear low-pass filter F whose impulse response is $g(\mu)$. In the continuous form this leads to:

$$\begin{aligned} x_{af}(\mu) = x_a(\mu) * g(\mu) &= [ax(\mu) + by(\mu)] * g(\mu) \\ &= ax(\mu) * g(\mu) + by(\mu) * g(\mu) = ax_f(\mu) + by_f(\mu) \end{aligned} \tag{10.83}$$

where $*$ denotes the convolution. Likewise,

$$y_{af}(\mu) = cx_f(\mu) + dy_f(\mu) \tag{10.84}$$

By comparison of equations (10.79-84), it is clear that point $(x_{af}(\mu), y_{af}(\mu))$ is the same as point $(x_f(\mu), y_f(\mu))$ transformed by the affine transform A. So curve $\Gamma_{af}(\mu)$ is the same as curve $\Gamma_{fa}(\mu)$. Theorem1 indicates that exchanging the computation order between affine transform and filtering does not change the result.

THEOREM2:

For any affine transform of a closed contour, using EAN sets parameter t to produce the curve $\Gamma_a(t)=(x_a(t), y_a(t))$. If area $s_p(t)$ is the area of an enclosed sector whose vertices are a pair of successive points and the centroid of the contour and if $\Gamma_{af}(t)=(x_{af}(t), y_{af}(t))$ indicates that $\Gamma_a(t)$ is filtered by a low-pass filter F, then the changes in enclosed areas $s_p(t)$ on the $\Gamma_{af}(t)$ are linear with affine mapping as illustrated on figure 10.39.

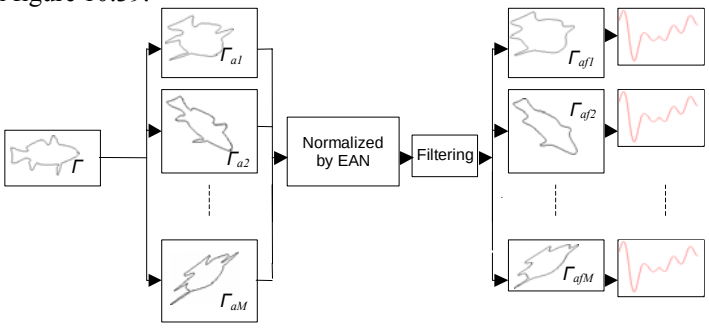


Figure 10.39: Illustration of theorem2

PROOF:

In section III, we know the enclosed area $s_p(t)$ of the triangle on the filtered affine contour whose vertices are $(x_{af}(t), y_{af}(t))$, $(x_{af}(t+1), y_{af}(t+1))$ and the centroid G is

$$s_p(t) = \frac{1}{2}\left|x_{af}(t)y_{af}(t+1) - x_{af}(t+1)y_{af}(t)\right| \tag{10.85}$$

Due to THEOREM1,

$$x_{af}(t) = x_{fa}(t) = ax_f(t) + by_f(t) \tag{10.86}$$

$$y_{af}(t) = y_{fa}(t) = cx_f(t) + dy_f(t) \tag{10.87}$$

and

$$x_{af}(t+1) = x_{fa}(t+1) = ax_f(t+1) + by_f(t+1) \tag{10.88}$$

$$y_{af}(t+1) = y_{fa}(t+1) = cx_f(t+1) + dy_f(t+1) \tag{10.89}$$

Therefore from equation (10.85), we can write

$$\begin{aligned} s_p(t) &= \frac{1}{2}\left|[ax_f(t) + by_f(t)][cx_f(t+1) + dy_f(t+1)]\right. \\ &\quad \left. -[ax_f(t+1) + by_f(t+1)][cx_f(t) + dy_f(t)]\right| \\ &= \frac{1}{2}\left|adx_f(t)y_f(t+1) + bcx_f(t+1)y_f(t)\right. \\ &\quad \left. - adx_f(t+1)y_f(t) - bcy_f(t+1)x_f(t)\right| \\ &= \frac{1}{2}\left|ad-bc\right| \cdot \left|x_f(t)y_f(t+1) - x_f(t+1)y_f(t)\right| \end{aligned} \tag{10.90}$$

Observing equation (10.90), $s_p(t)$ is just linearly proportional by a scale factor $\left|ad-bc\right|$. Accordingly we have proved that enclosed areas $s_p(t)$ are linear with affine mapping.

DEDUCTION:

The proportion $v'(t)$ of closed areas $s_p(t)$ with the total area S of the filtered contour is preserved under general affine transforms.

PROOF:

According to relation (10.90), the total area S of the filtered contour is:

$$S = \frac{1}{2}\left|ad-bc\right| \cdot \sum_{t=0}^{N-1}\left|x_f(t)y_f(t+1) - x_f(t+1)y_f(t)\right| \tag{10.91}$$

so that

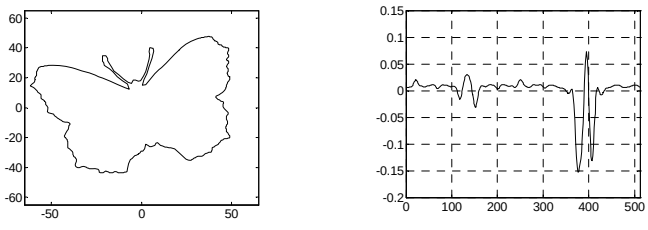
$$\begin{aligned} v'(t) &= s_p(t)/S \\ &= \left|x_f(t)y_f(t+1) - x_f(t+1)y_f(t)\right| / \sum_{t=0}^{N-1}\left|x_f(t)y_f(t+1) - x_f(t+1)y_f(t)\right| \end{aligned} \tag{10.92}$$

Equation (10.92) indicates that $v'(t)$ is not related to the affine parameters a, b, c and d. Therefore $v'(t)$ is preserved under general affine transforms. In addition, we can deduce a major property of $v'(t)$: the integration of

$v(t) = [v'(t) - 1/N]$ is equal to zero as shown by equation (10.93).

$$\sum_{t=0}^{N-1} v(t) = \sum_{t=0}^{N-1} [v'(t) - \frac{1}{N}] = \sum_{t=0}^{N-1} \frac{s_p(t)}{S} - \sum_{t=0}^{N-1} \frac{1}{N} = \frac{S}{S} - \frac{N}{N} = 0 \qquad (10.93)$$

We refer to vector *v(t)* as the normalized part area vector (NPAV). Figure 10.40 shows an example of NPAV. The contour is normalized to 512 points by EAN.



(a) **(b)**

Figure 10.40: An example of NPAV. (a) The contour of a butterfly. (b) The NPAV of the contour (a).

As THEOREM2 and its deduction show, in all cases, even those with severe deformations, the function $s_p(t)$ is also preserved. Only the amplitude changes under general affine transforms; the NPAV $v(t)$ has an affine-invariant feature. In the following section, we will present the results of our experiments which evaluate the property of the proposed algorithm.

9.4 Experimental results

In this section, we will evaluate the behaviour of NPAV $v(t)$ in relation to the affine transforms, EAN, filtering and noise by presenting various experimental results. We consider the results of our experiments on the MPEG-7 CE-shape-1 database containing 1400 images of shapes. The contours in this database yield a great range of shape variation. The framework of these experiments is shown in figure 10.41.

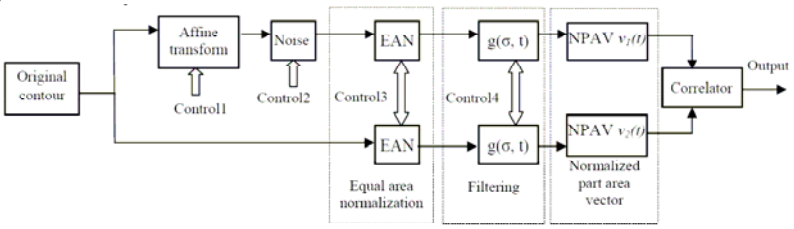


Figure 10.41: The framework of experiments

The input signal of the experiment is an original contour from the MPEG-7 database, and the output is the maximum linear correlation coefficient between the NPAV *v(t)* of the upper pathway and that of the lower pathway. The upper path-

way includes the affine transforms, noise power control, equal area normalization and the low-pass filter. The output signal of the upper pathway is the NPAV $v_1(t)$ of the affine contour. The lower pathway includes the equal area normalization and low-pass filter. The output signal of the lower pathway is the NPAV $v_2(t)$ of the original contour. The correlator is then applied to calculate the maximum linear correlation coefficient between NPAV $v_1(t)$ and NPAV $v_2(t)$ by shifting the vector $v_2(t)$ circularly.

The four control points are presented as follows:

- **Control1** is applied to control the parameters of affine transforms. We can control the affine contour by respectively scaling, rotating and shearing or mixing the transforms.
- **Control2** controls the power of the noise.
- **Control3** controls the parameters of equal area normalization in the upper and lower pathways. The parameters are the number of normalized points and the position of the starting point.
- **Control4** is applied to simultaneously control the bandwidth of the low-pass filters in the upper and lower pathways. Here, the filter is a Gaussian kernel with a scale parameter σ given by the standard deviation of the filter.

9.4.1 The NPAV and scaling transforms

The scaling transform is one of the affine transform modes. It is obtained by applying the matrix $A_{scaling}$ to the contour coordinates. For a uniform scaling transformation, we choose the parametric matrix

$$A_{Scaling1} = \begin{bmatrix} \gamma_1 & 0 \\ 0 & \gamma_1 \end{bmatrix} \tag{10.94}$$

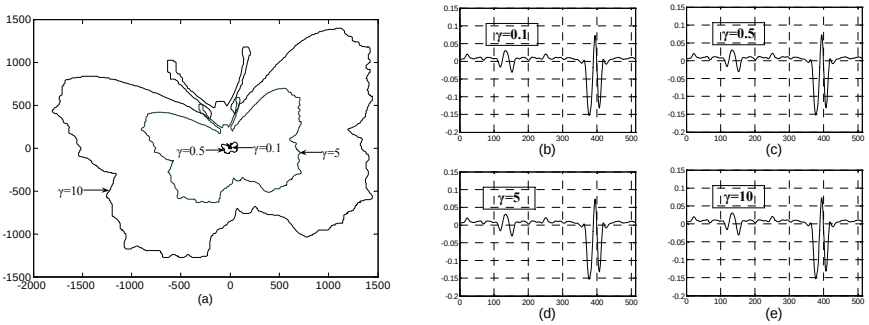


Figure 10.42: Illustration of the robustness of NPAV under uniform scaling transforms. (a) is 4 different scale contours. (b), (c), (d) and (e) are the NPAVs of each contour in (a) respectively.

Suppose γ_1 is successively equal to 0.1, 0.5, 5, 10, for the different observed contours of figure 10.42 (a): we obtain figures 10.42 (b)-(e) respectively. For the non-uniform scaling transformation, we choose the parametric matrix

$$A_{Scaling2} = \begin{bmatrix} \gamma_2 & 0 \\ 0 & 1 \end{bmatrix} \tag{10.95}$$

Suppose γ_2 is successively equal to 0.1, 0.5, 5, 10, for the figure 10.43 (a): we obtain figures 10.43 (b)-(e) respectively.

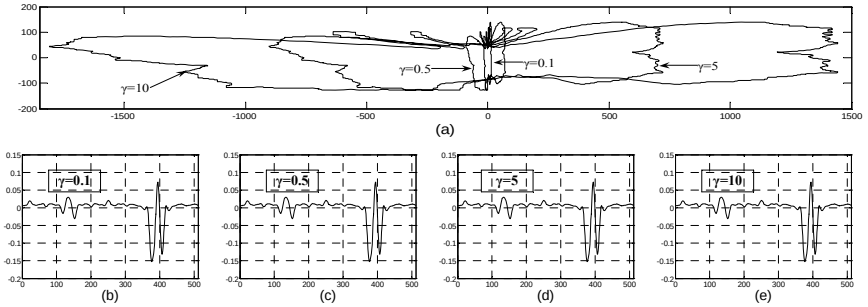


Figure 10.43: Illustration of the robustness of NPAV under non-uniform scaling transforms. (a) is 4 different scale contours. (b), (c), (d) and (e) are the NPAVs of each contour in (a) respectively.

Figures 10.42 and 10.43 show that although the shape of the butterfly changes with scaling, the NPAV remains identical.
Furthermore, we calculate statistical results. For all the shapes in the database, the average correlation between the NPAV of the original shape and that of its scaling transforms under matrixes $A_{scaling1}$ and $A_{scaling2}$ is presented in Table 10.2.

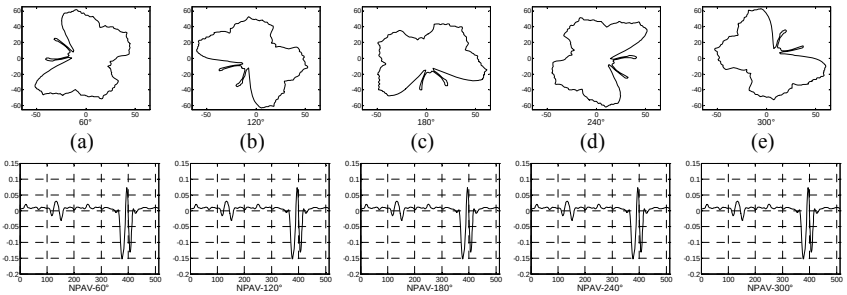
TABLE 10.2: CORRELATION WITH THE SCALING TRANSFORMS

	Correlation coefficient	
γ_1/γ_2	Uniform	Non-uniform
0.1	0.999	0.972
0.5	0.999	0.985
5	0.999	0.980
10	0.999	0.976

Table 10.2 shows that the uniform scaling transform has practically no effect on the NPAV of the contour and that non-uniform scaling affects the NPAV only a little. So the NPAV is very robust under scaling transforms.

9.4.2 The NPAV and rotation transforms

As stated before, a rotation transform is obtained by applying the matrix $A_{Rotation}$ to the contour coordinates. Let the rotation angles θ be 60°, 120°, 180°, 240° and 300°. The position of the starting point of a contour is unchanged. Figure 10.44 shows the rotated copies of the pattern on Figure 10.40(a) and their NPAVs. As can be seen in figure 10.44, all the NPAVs are identical. Furthermore, we present the statistical results. For all the shapes in the database, the average correlation between the NPAV of the original shape and that of its rotated transforms under the rotation angles θ is presented in Table 10.3.



(a) (b) (c) (d) (e)

(f) (g) (h) (i) (j)

Figure 10.44: Illustration of the robustness of NPAV under rotation: (a)-(e) are the same contour with 5 different orientations. (f)-(j) are the NPAVs of each contour in (a)-(e) respectively.

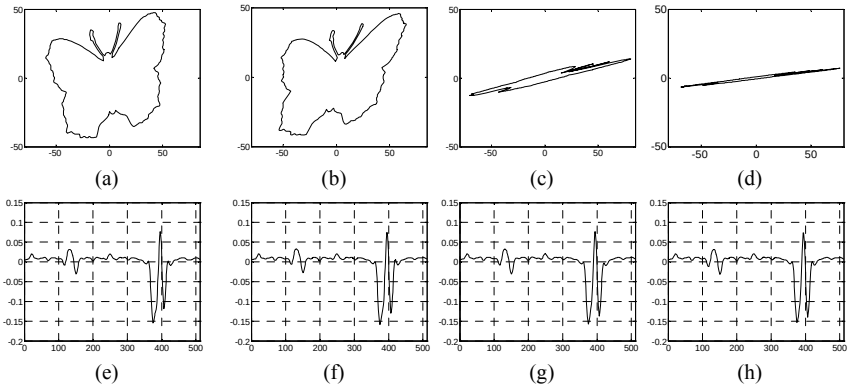
TABLE 10.3: CORRELATION WITH THE ROTATION TRANSFORMS

θ	Correlation coefficient
60°	1.000
120°	1.000
180°	1.000
240°	1.000
300°	1.000

In this way, as the position of the starting point does not change so that the NPAVs maintain their invariance.

9.4.3 The NPAV and shearing transforms

As introduced in subsection 9.1.2, a shearing is obtained by applying the matrix A_{Shear} to the contour coordinates. Let the shearing parameter k be successively equal to 0.1, 0.5, 5 and 10. Suppose the position of the starting point of the contour remains unchanged. Figure 10.45 shows the shearing copies of pattern figure 10.40(a) and their NPAVs.



(a) (b) (c) (d)

(e) (f) (g) (h)

Figure 10.45: Illustration of the robustness of NPAV under shearing transforms. (a)-(d) are the same contour with 4 different shearing transforms. (e)-(h) are the NPAVs of each contour in (a)-(d) respectively.

We further calculate the statistical results. For all the shapes in the database, the average correlation between the NPAV of the original shape and that of its shearing transform under shearing parameter k is presented in the Table 10.4.

TABLE 10.4: CORRELATION WITH THE SHEARING TRANSFORMS

k	Correlation coefficient
0.1	0.989
0.5	0.987
5	0.980
10	0.972

The three above experiments indicate that the descriptor NPAV is relatively affine-invariant. As shown below, we will evaluate the effect of equal area normalization (EAN) under the same affine transform. Suppose the contour rotates 60° counter clockwise and the shearing parameter k is 1.5. The affine matrix is

$$A = \begin{bmatrix} \frac{2+3\sqrt{3}}{4} & -\frac{3-2\sqrt{3}}{4} \\ \frac{\sqrt{3}}{2} & \frac{1}{2} \end{bmatrix} \tag{10.96}$$

Adjust 'Control2' to let the power of noise be 0. Adjust 'Control4' to let $\sigma = 10$. The contents of the experimental set include the two following aspects: the relation between the NPAV and the number of points normalized by EAN and the relation between the NPAV and the position of the starting points on a contour. The results are presented in subsections *9.4.4* and *9.4.5*.

9.4.4 The NPAV and the number of points normalized by EAN

The number of points on the contour is normalized to 64, 128 and 256. Figure 10.47 shows the original and the affined contours of the pattern in figure 10.41(a) with various numbers of points and their NPAVs.

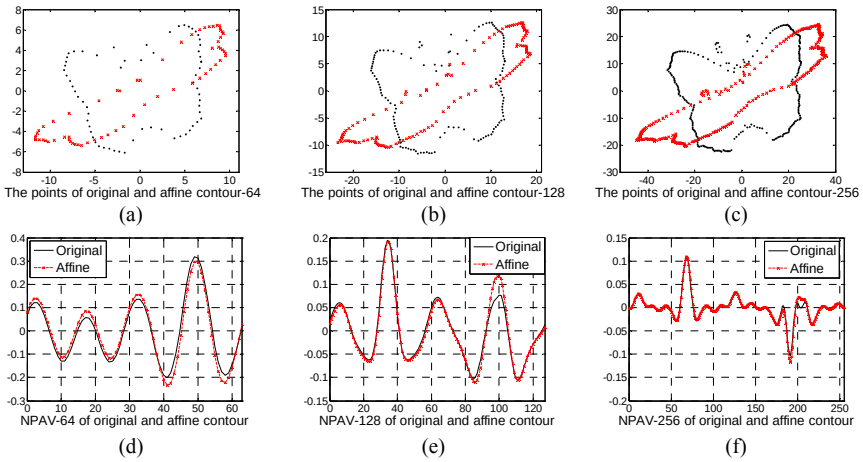


Figure 10.46: Illustration of the robustness of NPAV under the contour normalized to different number of points. (a)-(c) are the original and affine contours normalized to 64, 128 and 256 respectively. (d)-(f) are the NPAVs of the two contours in (a)-(c) respectively.

We notice that although the NPAVs of the contour with different numbers of points are very different from each other, the NPAV of the affine contour and that of original contour are almost identical under the same number of points normalized by EAN. Table 10.5 presents the statistical results. It shows that under the different number of points normalized by EAN, the NPAV changes slightly under affine transforms.

TABLE 10.5: CORRELATION UNDER DIFFERENT NORMALIZED NUMBER OF POINTS

Number of points	Correlation coefficient
64	0.985
128	0.992
256	0.993

9.4.5 The NPAV and the position of the starting point

Suppose the position of the starting point (SP) is located on the different positions '1', '2', '3' or '4', as illustrated on figure 10.47(a)-(d). These positions are located at 12.5%, 25%, 37.5% and 50% of the original starting with 100% corresponding to the total number of points. Suppose the contour is normalized to 256 points. figure 10.47 shows the effect of different starting points to the NPAV of the original and affine contours.

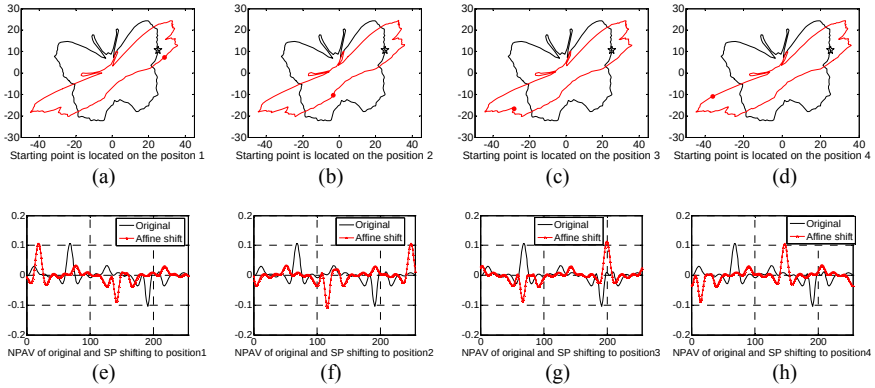


Figure 10.47: Illustration of the robustness of a NPAV with various positions of the starting point. (a)-(d) are the original and the affine contours with the starting point located at position 1-4 respectively. '☆'is the position of the starting point on the original contour; '●''is the position of the starting point on the affine contour . (e)-(h) are the NPAVs of the two contours with different positions for the starting point in (a)-(d) respectively.

As obvious from figure 10.47, the NPAV of various starting point positions are topologically identical except for a 'circular' delay. In this way, a shift in the starting point is equivalent to a circular delay in the NPAV.

We further calculate the statistical results. To calculate the correlation between the NPAV of the original contour and that of its affine transforms with a shift of starting point, we move the NPAV of the original shape point by point and search for

the highest value of the correlation. For all the shapes in the database, the average correlations of the various starting point positions under the same number of points normalized by EAN are presented in the Table 10.6.

As can be seen in the table, the position of the starting point on the contour does not affect the robustness of the NPAV.

TABLE 10.6: CORRELATION UNDER DIFFERENT POSITION OF STARTING POINT

Starting point shift	Correlation coefficient
12.5%	0.989
25%	0.987
37.5%	0.974
50%	0.972

9.4.6 The NPAV and noise

For different reasons, it happens that the curve undergoes perturbations so that it becomes noisy. To reduce the effect of noise, the curve is first smoothened by applying a low-pass Gaussian filter of standard deviation set to $\sigma = 2$. The NPAVs and their shape contaminated by the random uniform noise with different *SNR* are presented in figure 10.48. It reveals that the NPAV is quite robust to boundary noise and irregularities, even in the presence of severe noise. It is clear that, as the noise amplitude increases, the contours become more and more fuzzy. In order to calculate the average correlation coefficient, we do the experiments by contaminating the test contours with random uniform noise ranging from high to low *SNR* affecting the database.

Table 10.7 shows the average correlation coefficient of all the NPAVs of shapes in the database under different *SNR*.

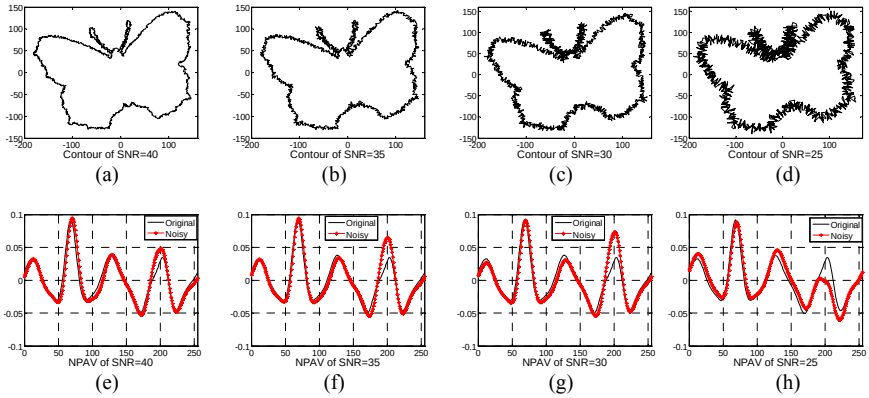


Figure 10.48: Demonstration of NPAV under the condition of different SNR. (a)-(d) are the contour contaminated by different noise power. (e)-(h) are the NPAVs of contours in (a)-(d) respectively.

This shows the NPAV's suitability for use in noisy conditions. By analyzing the experimental results, we notice that NPAV is quite robust to scale, orientation,

shearing of objects, noise and the position of starting point. Therefore, NPAV can be used to characterize a pattern for recognition purposes.

TABLE 10.7: CORRELATION UNDER DIFFERENT *SNR*

SNR	Correlation coefficient
40dB	0.964
35dB	0.963
30dB	0.949
25dB	0.898

9.4.7 Evaluation on pattern retrieval

In order to assess the retrieval performance, we create affine transformed versions of our existing shape contours. Suppose the contour rotates θ counter-clockwise, and the shearing parameter is k. The matrix A is then constructed as follows.

$$A = \begin{bmatrix} k \cdot \sin\theta + \cos\theta & k\cos\theta - \sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \tag{10.97}$$

Let us consider $\theta = 2n\pi / 5$ with n=0, 1, 2 and 4, and k=1 and 2. Therefore, 10 affine transforms are applied to each contour so that the new database consists of $1400 \times 10 = 14000$ transformed contours. The similarity measure between two NPAV attributes, *$v_1(i)$ and $v_2(i)$, i=0, 1, ..., N-1* $v_1(i), v_2(i), i \in [0, N-1]$ can be represented by the following functions:

$$d_1 = \min_{n=0}^{N-1}\left[\sum_{i=0}^{N-1}|v_1(i) - v_2(j)|\right] \qquad d_2 = \min_{n=0}^{N-1}\left[\sum_{i=0}^{N-1}|v_1(N-1-i) - v_2(j)|\right] \tag{10.98}$$

where $j = \begin{cases} i+n, & i+n < N \\ i+n-N, & i+n > N-1 \end{cases}$

Then similarity is given by:

$$d = \min(d_1, d_2) \tag{10.99}$$

Figure 10.49 gives examples of the retrieval results. Each example is presented in two rows, starting with the input query on the left-hand side and followed by the outputs from the system in response to that query. It denotes the first 20 retrieved contours and their similarity distance *d*.

We notice that all 10 affine transforms of the query contour appear in the first 10 images. And the similarity distance of the first non-relevant contour is much greater than that of related contours. So, we can retrieve the similar contours easily. For the 1400 query original contours, the statistical average distance of the first 10 related contours is only 31.5% of the average distance of the first non-relevant contour.

All these previous results indicate that NPAV stays robust under affine transforms.

	1.2367	1.4505	1.4572	1.8526	1.8854	2.2931	2.4610	2.6463	2.7881	4.1007
Fish-15	5.5887	5.6306	5.6731	5.6792	5.7004	5.7073	5.7386	5.7572	5.7661	5.7833
	2.3989	3.1920	3.2550	3.4063	3.4355	3.6983	4.7405	4.9794	5.1912	5.9434
Bat-15	17.5724	17.5750	17.6659	17.7552	17.8595	17.9258	17.9436	17.9798	18.0819	18.1035
	2.3200	2.5564	2.9929	3.0492	3.3157	3.4922	3.7418	4.7902	5.2911	5.9047
Bird-16	12.1847	12.2113	12.2150	12.2740	12.3852	12.3920	12.4190	12.4462	12.4556	12.4789
	2.4890	2.5587	2.7941	3.2032	3.2607	3.5586	4.0831	5.2968	5.7779	6.7085
Guitar-02	8.2862	8.3074	8.3220	8.3325	8.3420	8.3433	8.3543	8.3818	8.4336	8.5227

Figure 10.49: Illustrative retrieval results obtained by the multi-scale NPAV.

10 Conclusion

In this chapter we have studied and compared the methods of shape-based feature extraction and representation. About 40 techniques for extraction of shape features have been shortly described, referenced in a bibliography and syntheticaly compared. Unlike the traditional classification, the approaches of shape-based feature extraction and representation were classified by their processing approaches. These processing approaches included shape signatures, polygonal approximation methods, spatial interrelation feature, moments approaches, scale-space methods and shape transform domains: in such way, one can easily select the appropriate processing approach. A synthetic table has been established for a fast and global comparison of the performances of these approaches.

To go more deeply in shape based feature extraction we have also described and evaluated a new method designed for extracting invariants of a shape under affine transform. Our representation is based on the association of two parameters: the affine arc length and the enclosed area, viz. we normalize a contour to affine-invariant length by the affine enclosed area. For the needs of this new approach,

we proved two theorems and a deduction. They revealed that, for a filtered contour, the part enclosed area is linear under affine transforms. We further defined the affine-invariance vector: the normalized part area vector (NPAV). After a number of experiments applied to the MPEG-7 CE-shape-1 database, we demonstrated that NPAV is quite robust with respect to affine transforms and noise, even in the presence of severe noise.

References

[1] Abbasi, S.; Mokhtarian, F. & Kittler, J. (2000), 'Enhancing CSS-based shape retrieval for objects with shallow concavities', *Image and Vision Computing* **18(3)**, 199-211.

[2] Alajlan, N.; Kamel, M. S. & Freeman, G. (2006), 'Multi-object image retrieval based on shape and topology', *Signal Processing: Image Communication* **21**, 904-918.

[3] Alajlan, N.; Rube, I. E.; Kamel, M. S. & Freeman, G. (2007), 'Shape retrieval using triangle-area representation and dynamic space warping', *Pattern Recognition* **40(7)**, 1911-1920 .

[4] Arbter, K.; Snyder, W.; Burkhardt, H. & Hirzinger, G. (1990), 'Applications of affine-invariant Fourier descriptors to recognition of 3-Dobjects', *IEEE Trans. Pattern Analysis and Machine Intelligence* **12(7)**, 640-646.

[5] Arica, N. & Vural, F. (2003), 'BAS: a perceptual shape descriptor based on the beam angle statistics', *Pattern Recognition Letters* **24(9-10)**.

[6] Badawy, O. E. & Kamel, M. (2004), Shape Retrieval using Concavity Trees, *in* 'Proceedings of the 17th International Conference on Pattern Recognition', pp. 111-114.

[7] Bauckhage, C. & Tsotsos, J. K. (2005), Bounding box splitting for robust shape classification, *in* 'Proc. IEEE International Conference on Image Processing', pp. 478-481.

[8] Belongie, S.; Malik, J. & Puzicha, J. (2002), 'Shape Matching and Object Recognition Using Shape Context', *IEEE Trans. Pattern Analysis and Machine Intelligence* **24(4)**, 509-522.

[9] Berretti, S.; Bimbo, A. D. & Pala, P. (2000), 'Retrieval by shape similarity with perceptual distance and effective indexing', *IEEE Trans. on Multimedia* **2(4)**, 225-239.

[10] Celebi, M. E. & Aslandogan, Y. A. (2005), A Comparative Study of Three Moment-Based Shape Descriptors, *in* 'Proc. of the International Conference of Information Technology: Codingand Computing', pp. 788-793.

[11] Chakrabarti, K.; Binderberger, M.; Porkaew, K. & Mehrotra, S. (2000), Similar shape retrieval in MARS, *in* 'Proc. IEEE International Conference on Multimedia and Expo '.

[12] Chen, G. & Bui, T. D. (1999), 'Invariant Fourier-wavelet descriptor for pattern recognition', *Pattern Recognition* **32**, 1083-1088.

[13] Chuang, C.-H. & Kuo, C.-C. (1996), 'Wavelet Descriptor of Planar Curves: Theory and Applications', *IEEE Trans. Image Processing* **5(1)**, 56-70.

[14] Davies, E. (1997), *Machine Vision: Theory, Algorithms, Practicalities*, Academic Press, New York.

[15] Dubinskiy, A. & Zhu, S. C. (2003), A Multi-scale Generative Model for Animate Shapes and Parts, *in* 'Proc. Ninth IEEE International Conference on Computer Vision (ICCV)'.

[16] Gonzalez, R. & Woods, R. (2002), *Digital image processing, Second Edition*, PEARSON EDUCATION NORTH ASIA LIMITED and Publishing House of Electronics Industry.

[17] Guru, D. & Nagendraswam, H. (2007), 'Symbolic representation of two-dimensional shapes', *Pattern Recognition Letters* **28**, 144-155.

[18] Han, S. & Yang, S. (2005), An Invariant Feature Representation for shape Retrieval, *in* 'Proc. Sixth International Conference on Parallel and Distributed Computing, Applica-

tions and Technologies '.
[19] Jalba, A.; Wilkinson, M. & Roerdink, J. (2006), 'Shape representation and recognition through morphological curvature scale spaces', *IEEE Trans. Image Processing* **15(2)**, 331-341.
[20] Jin, K.; Cao, M.; Kong, S. & Lu, Y. (2006), Homocentric Polar-Radius Moment for Shape Classification, *in* 'Proc. Signal Processing, The 8th International Conference on'.
[21] Kan, C. & Srinath, M. D. (2002), 'Invariant character recognition with Zernike and orthogonal Fourier-Mellin moments', *Pattern Recognition* **35**, 143-154.
[22] Kauppinen, H.; Seppanen, T. & Pietikainen, M. (1995), 'An Experimental Comparison of Auto-regressive and Fourier-Based Descriptors in 2-D Shape Classification', *IEEE Trans. Pattern Analysis and Machine Intelligence* **17(2)**, 201-207.
[23] Khalil, M. & Bayoumi, M. (2001), 'A Dyadic Wavelet Affine Invariant Function for 2D Shape Recognition', *IEEE Trans. Pattern Analysis and Machine Intelligence* **25(10)**, 1152-1164.
[24] Kpalma, K. & Ronsin, J. (2006), 'Multiscale contour description for pattern recognition', *Pattern Recognition Letters* **27**, 1545-1559.
[25] Latecki, L. J. & Lakamper, R. (2000), 'Shape Similarity Measure Based on Correspondence of Visual Parts', *IEEE Trans. Pattern Analysis and Machine Intelligence* **22(10)**, 1185-1190.
[26] Latecki, L. J. & Lakamper, R. (1999), 'Convexity rule for shape decomposition based on discrete Contour Evolution', *Computer Vision and Image Understanding* **73(3)**, 441-454.
[27] Lee, S.-M.; Abbott, A. L.; Clark, N. A. & Araman, P. A. (2006), A Shape Representation for Planar Curves by Shape Signature Harmonic Embedding, *in* 'Proc. IEEE Computer Society Conference on Computer Vision and Pattern Recognition'.
[28] Liu, Y. K.; Wei, W.; Wang, P. J. & Zalik, B. (2007), 'Compressed vertex chain codes', *Pattern Recognition* **40(11)**, 2908-2913.
[29] Lu, G. & Sajjanhar, A. (1999), 'Region-based shape representation and similarity measure suitable for content based image retrieval', *ACM Multimedia System Journal* **7(2)**, 165-174.
[30] Lu, K.-J. & Kota, S. (2002), Compliant Mechanism Synthesis for Shape-Change Applications: Preliminary Results, *in* 'Proceedings of SPIE Modelling, Signal Processing, and Control Conference', pp. 161-172.
[31] Mehtre, B. M.; Kankanhalli, M. S. & Lee, W. F. (1997), 'Shape Measures for Content Based Image Retrieval: A Comparison', *Pattern Recognition* **33(3)**, 319--337.
[32] Mokhtarian, F. & Mackworth, A. K. (1992), 'A Theory of Multiscale, Curvature-Based Shape Representation for Planar Curves', *IEEE Trans. Pattern Analysis and Machine Intelligence* **14(8)**, 789-805.
[33] Mori, G. & Malik, J. (2002), Estimating human body configurations using shape context matching, *in* 'Proc. 7th European Conference on Computer Vision', pp. 666-680.
[34] Mukundan, R. (2004), A new class of rotational invariants using discrete orthogonal moments, *in* 'Sixth IASTED International Conference on Signal and Image Processing', pp. 80-84.
[35] Mukundan, R.; Ong, S. & Lee, P. (2001), 'Image Analysis by Tchebichef Moments', *IEEE Trans. Image Processing* **10(9)**, 1357-1364.
[36] Peng, J.; Yang, W. & Cao, Z. (2006), A Symbolic Representation for Shape Retrieval in Curvature Scale Space, *in* 'Proc. International Conference on Computational Intelligence for Modelling Control and Automation and International Conference on Intelligent Agents Web Technologies and International Commerce'.
[37] Ricard, J.; Coeurjolly, D. & Baskurt, A. (2005), 'Generalizations of angular radial transform for 2D and 3D shape retrieval', *Pattern Recognition Letters* **26(14)**.
[38] Sebastian, T.; Klein, P. & Kimia, B. (2004), 'Recognition of Shapes by Editing Their

Shock Graphs', *IEEE Trans. Pattern Analysis and Machine Intelligence* **26(5)**, 550-571.
[39] Siddiqi, K. & Kimia, B. (1996), A Shock Grammar for Recognition, *in* 'Proceedings of the IEEE Conference Computer Vision and Pattern Recognition', pp. 507-513.
[40] Smith, S. P. & Jain, A. K. (1982), 'Chord distribution for shape matching', *Computer Graphics and Image Processing* **20**, 259-271.
[41] Sonka, M.; Hlavac, V. & Boyle, R. (1993), *Image Processing, Analysis and Machine Vision*, Chapman and Hall, London, UK.
[42] Tabbone, S.; Wendling, L. & Salmon, J.-P. (2006), 'A new shape descriptor defined on the Radon transform', *Computer Vision and Image Understanding* **102(1)**, 42-51.
[43] Taubin, G. & Cooper, D. (1991), Recognition and positioning of rigid objects using algebraic moment invariants, *in* 'SPIE Conference on Geometric Methods in Computer Vision', pp. 175-186.
[44] Taza, A. & Suen, C. (1989), 'Discrimination of planar shapes using shape matrices', *IEEE Trans. System, Man, and Cybernetics* **19(5)**, 1281-1289.
[45] Thayananthan, A.; Stenger, B.; Torr, P. H. S. & Cipolla, R. (2003), Shape Context and Chamfer Matching in Cluttered Scenes, *in* 'Proc. IEEE Computer Society Conference on Computer Vision and Pattern Recognition'.
[46] Tieng, Q. M. & Boles, W. W. (1997), 'Wavelet-Based Affine Invariant Representation: A Tool for Recognizing PlanarObjects in 3D Space', *IEEE Trans. Pattern Analysis and Machine Intelligence* **19(8)**, 846-857.
[47] Wang, Y. P.; Lee, K. T. & K.Toraichi (1999), 'Multiscale curvature-based shape representation using B-spline wavelets', *IEEE Trans. Image Process* **8(10)**, 1586-1592.
[48] Yadava, R. B.; Nishchala, N. K.; Gupta, A. K. & K.Rastogi, V. (2007), 'Retrieval and classification of shape-based objects using Fourier, generic Fourier, and wavelet-Fourier descriptors technique: A comparative study', *Optics and Lasers in Engineering* **45(6)**, 695-708.
[49] Yang, M.; Kpalma, K. & Ronsin, J. (2007), Scale-controlled area difference shape descriptor, *in* 'Proc. SPIE, Electronic Imaging science and Technology'.
[50] Zahn, C. T. & Roskies., R. Z. (1972), 'Fourier Descriptors for Plane closed Curves', *IEEE Trans. Computer* **c-21(3)**, 269-281.
[51] Zhang, D. & Lu, G. (2004), 'Review of shape representation and description techniques', *Pattern Recognition* **37**, 1-19.
[52] Zhang, D. & Lu, G. (2003), 'A comparative study of curvature scale space and Fourier descriptors for shape-based image retrieval', *Visual Communication and Image Representation* **14(1)**.
[53] Zhang, D. & Lu, G. (2002), A Comparative Study of Fourier Descriptors for Shape Representation and Retrieval, *in* 'Proc. 5th Asian Conference on Computer Vision'.
[54] Zhang, D. S. & Lu, G. (2001), A Comparative Study on Shape Retrieval Using Fourier Descriptors with DifferentShape Signatures, in 'Proc. International Conference on Intelligent Multimedia and Distance Education(ICIMADE01)'
[55] Zhang, H. & Malik, J. (2003), Learning a discriminative classifier using shape context distances, *in* 'Proc. IEEE Computer Society Conference on Computer Vision and Pattern Recognition'.
[56] ISO/IEC JTC1/SC29/WG11, (2004), 'MPEG-7 Overview (version 10) ', Technical report.
[57] Hu, M.-K. (1962), 'Visual Pattern Recognition by Moment Invariants', IRE Trans. Information Theory IT-8, 179-187.
[58] Iivarinen, J. & Visa, A. (1996), Shape recognition of irregular objects, in 'Proc. SPIE, Intelligent Robots and Computer Vision XV: Algorithms, Techniques, Active Vision, and Materials Handling', pp. 25-32.
[59] Flusser, J. (1992), Invariant Shape Description and Measure of Object Similarity, in

'Proc. 4th International Conference on Image Processing and its Applications', pp. 139-142.

[60] di Baja, G. S. & Thiel, E. (1996), 'Skeletonization algorithm running on path-based distance maps', Image Vision Computer 14, 47-57.

[61] Borgefors, G. (1986), Distance Transformations in Digital Images, in 'Computer Vision, Graphics, and Image Processing', pp. 344-371.

[62] Kolesnikov, A. (2003), Efficient algorithms for vectorization and polygonal approximation, Ph.D thesis, University of Joensu, Finland